OParl 1.1

Spezifikation einer einheitlichen Schnittstelle zum Abruf von maschinenlesbaren Informationen aus Ratsinformationssystemen.

Version v1.1.1-23-g0ca3435
20.06.2018

Inhaltsverzeichnis

1 Einleitung

Dieses Dokument enthält die Spezifikation des OParl Schnittstellen-Standards für parlamentarische Informationssysteme[1]. Es dient damit als Grundlage für die Implementierung von OParl-konformen Server- und Clientanwendungen.

1.1 Was ist OParl?

OParl ist die Gruppierung, die Initiator und Herausgeber der vorliegenden Spezifikation ist. An OParl wirken Verbände, zivilgesellschaftliche Organisationen und Initiativen und Softwareanbieter sowie interessierte Einzelpersonen mit.

Die vorliegende Spezifikation beschreibt den OParl-Standard. Dieser definiert eine Webserviceschnittstelle, die den anonymen, lesenden Zugriff auf öffentliche Inhalte aus parlamentarischen Informationssystemen ermöglicht. Wie der Name "Webservice" ausdrückt, setzt diese Schnittstelle auf dem World Wide Web auf. Sie ermöglicht, dass parlamentarische Informationen maschinenlesbar als offene Daten (Open Data) veröffentlicht werden.

Die vorliegende Version ist die erste verabschiedete Version der Spezifikation des OParl-Standards.

1.2 Zielsetzung von OParl

OParl richtet sich an verschiedene Nutzergruppen und Stakeholder:

- Verwaltungen und andere politische Gremien in Gebietskörperschaften
- Bürger, politische Parteien und Organisationen
- Open-Data-Initiativen
- Wissenschaftler
- Anbieter von Server- und Softwareprodukten im Umfeld von parlamentarischen Informationssystemen und Öffentlichkeitsbeteiligung

Die Gründe, warum Betreiber von parlamentarischen Informationssystemen den Zugriff darauf über eine standardisierte Schnittstelle ermöglichen sollten oder möchten, können vielfältig und je nach Nutzergruppe unterschiedlich sein.

Ein zentrales Argument für Verwaltung und politische Gremien, sei es in Gebietskörperschaften oder auf Landes- oder Bundesebene, ist die Verpflichtung der Parlamente gegenüber der Bevölkerung, diese über die Fortschritte der parlamentarischen Arbeit zu informieren und auf dem Laufenden zu halten. Ein erster Schritt, der Bevölkerung Einblicke in die Arbeit und Zugriff auf Dokumente zu gewähren, ist vielerorts in den letzten Jahren durch Einführung von Ratsinformationssystemen mit anonymem, lesendem Zugriff über das World Wide Web gemacht worden.

Die damit eingeschlagene Richtung konsequent weiter zu gehen, bedeutet, die Daten der parlamentarischen Informationssysteme soweit offen zu legen, wie die Inhalte es erlauben. Es bedeutet, die Daten und Inhalte so universell weiterverwendbar und so barrierearm wie möglich anzubieten, dass jegliche weitere Verwendung durch Dritte technisch möglich ist. Der seit einiger Zeit etablierte Begriff für dieses Prinzip heißt "Open Data".

[1] In Deutschland hat sich auf kommunaler Ebene der Begriff "Ratsinformationssystem" etabliert. OParl ist jedoch nicht auf Gemeinderäte beschränkt und verwendet daher den Begriff "parlamentarisches Informationssystem".

Open-Data-Initiativen können unter Rückgriff auf Ratsinformationssysteme (RIS) mit OParl-Schnittstelle einfacher Dokumente und Daten aus unterschiedlichen Gebietskörperschaften in Open-Data-Katalogen verzeichnen und so einfacher auffindbar machen für die Weiterverwendung durch Dritte.

Bürgerinnen und Bürger, politische Parteien und zivilgesellschaftliche Organisationen können einfacher auf Inhalte parlamentarischer Informationssysteme zugreifen und diese entsprechend ihren Interessen aufbereiten. Dies können beispielsweise Visualisierungen von enthaltenen Daten, die Anreicherung von Informationsangeboten für spezielle Nutzergruppen oder die Schaffung von Benutzeroberflächen mit besonderen Funktionen für verschiedene Endgeräte sein.

Das Interesse an parlamentarischen Informationen und an Anwendungen, die diese nutzbar und auswertbar machen, ist offensichtlich vorhanden. Die Entwickler der alternativen Ratsinformationssysteme wie Frankfurt Gestalten[2], Offenes Köln oder der OpenRuhr:RIS-Instanzen (die letzten beiden wurden zusammengeführt in Politik Bei Uns[3]) wissen zu berichten, wie viel Interesse den Projekten gerade aus Orten entgegen gebracht wird, in denen derartige Systeme noch nicht verfügbar sind.

Die Anwendungsmöglichkeiten für parlamentarische Informationen, wenn sie über eine Schnittstelle schnell und einfach abgerufen werden können, sind vielfältig. Beispiele sind:

- Apps für den Abruf auf mobilen Endgeräten
- Möglichkeiten zur Wiedergabe für Nutzerinnen und Nutzer mit Beeinträchtigung des Sehvermögens
- Alternative und erweiterte Suchmöglichkeiten in Inhalten
- Auswertung und Analyse von Themen, Inhalten, Sprache etc.
- Benachrichtigungsfunktionen beim Erscheinen bestimmter Inhalte

Die Standardisierung dieses Zugriffs über die Grenzen einzelner Systeme hinweg erlaubt zudem, diese Entwicklungen auch geographisch und politisch grenzüberschreitend zu denken. Damit steigt nicht nur die potenzielle Nutzerschaft einzelner Entwicklungen. Auch das Potenzial für Kooperationen zwischen Anwendungsentwicklern wächst.

Für Wissenschaftler, die z. B. an vergleichenden Untersuchungen zu Vorgängen in verschiedenen Gebietskörperschaften interessiert sind, ergeben sich ebenso vielfältige Möglichkeiten über mehrere RIS-Instanzen hinweg auf entsprechende Informationen zuzugreifen und diese so einfacher in ihre Analysen einzubeziehen.

Darüber hinaus sind auch Motivationen innerhalb von Organisationen und Körperschaften erkennbar. So sollen parlamentarische Informationssysteme vielerorts in verschiedenste Prozesse und heterogene Systemlandschaften integriert werden. Durch eine einheitliche Schnittstelle bieten sich effiziente Möglichkeiten zur Integration der Daten in anderen Systeme, wie beispielsweise Webportale.

Anbieter von Softwareprodukten, die RIS-Lösungen anbieten, können ihren Kunden mit der Implementation der OParl-Schnittstelle eine entsprechende einheitliche Schnittstelle anbieten.

Ausführlichere Beschreibungen einiger möglicher Anwendungsszenarien finden sich im Kapitel Nutzungsszenarien.

1.3 Transparenz und Beteiligung durch Open Data

Öffentliche Stellen verfügen über vielfältige Informationen und Daten. Seit einigen Jahren sind zivilgesellschaftliche Organisationen sowie Politik und Verwaltung unter dem Schlagwort *Open*

[2]Frankfurt Gestalten: http://www.frankfurt-gestalten.de/
[3]Politik Bei Uns: https://politik-bei-uns.de

Data international und auch in Deutschland um eine stärkere Öffnung dieser Daten bemüht[4]. Bei dem Ansatz Open Data[5] geht es darum, diese Daten so bereitzustellen, dass Dritte diese einfacher finden und weiterverwenden können.

Die zehn Open-Data-Prinzipien der Sunlight-Foundation[6] beschreiben die Offenheit von Datensätzen. Wesentlich dabei sind vor allem die einfache rechtliche und die technische Offenheit. Bei ersterer geht es darum, dass Datensätze unter Nutzungsbestimmungen bereitgestellt werden, die kurz und verständlich formuliert sind und mindestens jegliche weitere Verwendung inklusive der kommerziellen erlauben, unter der Voraussetzung, dass bei der Weiterverwendung die Quelle benannt wird. Bei der technischen Offenheit steht die Bereitstellung von Datensätzen in möglichst maschinenlesbaren Formaten im Vordergrund. Dies bedeutet, stärker strukturierte Datensätze sind in der Bereitstellung zu bevorzugen. Liegen Daten innerhalb einer Organisation in einer Datenbank vor, so bietet es sich an, diese über eine Programmierschnittstelle (API) für Außenstehende bereitzustellen.

Die Erfüllung dieser rechtlichen und technischen Offenheit erlaubt es im Falle von OParl Dritten – dies können Bürgerinnen und Bürger, Unternehmen, Forschungseinrichtungen oder auch andere Verwaltungseinheiten sein – die Verwaltungsdaten wesentlich unkomplizierter für eigene Vorhaben wie Anwendungen oder Visualisierungen einzusetzen. Mit dem Ansatz offener Verwaltungsdaten soll so erstens mehr Transparenz über Prozesse und Entscheidungen in Politik und Verwaltung erreicht werden. Zweitens können Dritte auf Grundlage dieser Daten leichter eigene Geschäftsmodelle verfeinern oder neue entwickeln. Drittens wird es auch öffentlichen Stellen selbst erleichtert bereits im öffentlichen Sektor existierende Daten zu finden und weiterzuverwenden.

Das Prinzip offener Daten bzw. offener Verwaltungsdaten über die Minimalprinzipien rechtlicher und technischer Offenheit hinaus in die Tat umzusetzen, erfordert im Einzelfall häufig eine Zusammenarbeit von Datenbereitstellern und potentiellen Datennutzern. Die bloße Bereitstellung einer OParl-konformen API wird weder die Einhaltung der technischen Prinzipien, noch der weiteren Open-Data-Prinzipien vollständig garantieren. Viele Bestandteile der OParl-Spezifikation, die einen weitgehend barrierearmen Zugang zu Informationen[7] ermöglichen sollen, sind in der vorliegenden Version noch optional (Beispiel: Volltexte von Dokumenten über die API abrufbar machen). Andere Bestandteile, die von Interesse wären, sind noch gar nicht von OParl abgedeckt (Beispiel: Abstimmungsergebnisse). Grund dafür ist, dass sich OParl in einem frühen Stadium befindet und primär am Status Quo der parlamentarischen Informationssysteme ausgerichtet ist. Es liegt also auch weiterhin an Verwaltung und Politik, durch einen verantwortungsvollen Umgang mit den Systemen die maximal erreichbare Transparenz zu bieten. Das fängt bei verfügbaren Dokumentformaten an (ein PDF mit digitalem Text weist weit weniger Barrieren auf, als ein gescannter Brief, der ebenfalls als PDF gespeichert wurde) und hört bei der verwendeten Sprache auf[8].

[4] Eine weltweite Übersicht zu Open-Data-Projekten bietet z. B. der Open-Data-Showroom http://opendata-showroom.org/de/

[5] vgl. https://de.wikipedia.org/wiki/Open_data

[6] Ten Principles for Opening Up Open Government Information, https://sunlightfoundation.com/policy/documents/ten-open-data-principles

[7] Barrierefreie Informationstechnik-Verordnung 2.0 http://www.gesetze-im-internet.de/bitv_2_0/

[8] Weitere generelle Informationen zur Bereitstellung offener Verwaltungsdaten bieten bspw.

- Praktische Informationen: Open-Data-Handbook der Open Knowledge Foundation http://opendatahandbook.org/de/how-to-open-up-data/index.html
- Grundsätzliche Informationen: Die vom Bundesministerium des Innern beauftragte Studie "Open Government Data Deutschland" http://www.bmi.bund.de/SharedDocs/Downloads/DE/Themen/OED_Verwaltung/ModerneVerwaltung/opengovernment.pdf

1.4 Nutzungsszenarien

Für OParl sind verschiedene Nutzungsszenarien denkbar. Die Nachfolgende Auflistung soll einen kleinen Überblick geben, erhebt aber bei weitem keinen Anspruch auf Vollständigkeit:

1.4.1 Mobile Anwendung

Eine Anwendung für mobile Endgeräte wie Smartphones und Tablets, nachfolgend "App" genannt, könnte das Ziel verfolgen, Nutzern unterwegs, sowie abseits vom Desktop-PC optimierten Zugriff auf ein parlamentarisches Informationssystem zu ermöglichen. Dies könnte auch zur Vereinfachung der bisherigen Prozesse beitragen, da Nutzerinnen z.B. die Möglichkeit gegeben werden kann, auf Einladungen zu reagieren, oder Protokolle zu lesen.

1.4.2 Integration in ein Webportal

Portallösungen bieten den Betreibern die Möglichkeit, Inhalte auf einer einheitlichen Weboberfläche zu veröffentlichen, die aus verschiedensten Quellen und Plattformen bereitgestellt werden. Ein Beispiel für die Realisierung eines solchen Integrations-Ansatzes wäre eine Kommune, die für ihre allgemeine Website eine Portallösung einsetzt und hier auch Inhalte aus dem kommunalen Ratsinformationssystem einspeisen und darstellen möchte. Vorteil einer solchen Einbindung, also der kontextbezogenen Darstellung von parlamentarischen Informationen im Gegensatz zu einem monolithischen parlamentarischen Informationssystem könnte sein, dass Nutzer in einer gewohnten und akzeptierten Oberfläche jeweils die relevanten Informationen erhalten, ohne sich an die ungewohnte Umgebung eines parlamentarischen Informationssystems gewöhnen zu müssen.

1.4.3 Meta-Suche

Die Ermöglichung einer nutzerfreundlichen Suche, die damit verbundene Indexierung von verschiedensten Dokumenteninhalten und die Kategorisierung von Inhalten kann eine sowohl konzeptionell als auch technisch anspruchsvolle Aufgabe sein. Angelehnt an das seit den Anfängen des Webs etablierte Modell der externen Web-Suchmaschine sind spezielle Suchmaschinen für OParl-konforme parlamentarische Informationssysteme denkbar. Solche Plattformen treten gegenüber dem OParl-Server als Client auf und rufen bestimmte oder sämtliche Informationen, die das System bereithält, ab. Vorbild sind die Robots oder Spider von Web-Suchmaschinen. Die abgerufenen Informationen können dann indexiert und je nach Anforderungen für eine gezielte Suche weiterverarbeitet werden.

1.4.4 Forschungsprojekt "Themenanalyse"

In einem Forschungsprojekt sollen Pro- und Contra-Argumentationen bei Ratsdiskussionen zum Ausbau von Stromtrassen identifiziert werden. Dazu nutzen die Mitarbeitenden des Forschungsprojektes die OParl-Schnittstellen der parlamentarischen Informationssysteme aller Kommunen entlang der geplanten überregionalen Trassen. Über diese einheitlichen Schnittstellen können sie insbesondere die relevanten Wortprotokolle abrufen und zum Beispiel in einem Werkzeug zur qualitativen Datenanalyse lokal verarbeiten.

1.5 Nomenklatur

1.5.1 Zwingende, empfohlene und optionale Anforderungen

Diese Spezifikation nutzt **müssen**, **können** und **sollten** in einer eindeutig definierten Art und Weise. Diese ist angelehnt an die Definitionen der Begriffe MUST, SHOULD und MAY (bzw. MUST NOT, SHOULD NOT und MAY NOT) aus RFC2119.[9]

Die Bedeutung im Einzelnen:

müssen/muss bzw. zwingend: Die Erfüllung einer so gekennzeichneten Anforderung ist zwingend erforderlich.

Die Entsprechung in RFC2119 lautet “MUST”, “REQUIRED” oder “SHALL”.

nicht dürfen/darf nicht: Dieses Stichwort kennzeichnet ein absolutes Verbot.

Die Entsprechung in RFC2119 lautet “MUST NOT” oder “SHALL NOT”.

sollten/sollte bzw. empfohlen: Mit dem Wort **sollten** bzw. **sollte** sind empfohlene Anforderungen gekennzeichnet, die von jeder Implementierung erfüllt werden sollten. Eine Nichterfüllung ist als Nachteil zu verstehen, beispielsweise weil die Nutzerfreundlichkeit dadurch Einbußen erleidet, und sollte daher sorgfältig abgewogen werden.

Die Entsprechung in RFC2119 lautet “SHOULD” oder “RECOMMENDED”.

sollten nicht/sollte nicht bzw. nicht empfohlen: Diese Formulierung wird verwendet, wenn unter gewissen Umständen Gründe existieren können, die ein bestimmtes Verhalten akzeptabel oder sogar nützlich erscheinen lassen, jedoch die Auswirkung des Verhaltens vor einer entsprechenden Implementierung verstanden und abgewogen werden sollten.

Die Entsprechung in RFC2119 lautet “SHOULD NOT” oder “NOT RECOMMENDED”.

dürfen/darf bzw. optional: Mit dem Wort **dürfen** bzw. **darf** oder **optional** sind optionale Bestandteile gekennzeichnet. Ein Anbieter könnte sich entscheiden, den entsprechenden Bestandteil aufgrund besonderer Kundenanforderungen zu unterstützen, während andere diesen Bestandteil ignorieren könnten. Implementierer von Clients oder Servern **dürfen** in solchen Fällen **nicht** davon ausgehen, dass der jeweilige Kommunikationspartner den entsprechenden, optionalen Anteil unterstützt.

Die Entsprechung in RFC2119 lautet “MAY”.

1.5.2 Geschlechterspezifische Begrifflichkeiten

Um bei Begriffen wie Nutzer, Anwender, Betreiber etc. die sonst übliche Dominanz der männlichen Variante zu vermeiden, werden in diesem Dokument männliche und weibliche Varianten gemischt. Gemeint sind in allen Fällen Personen jeglichen Geschlechts.

1.5.3 Codebeispiele

Die in diesem Dokument aufgeführten Codebeispiele dienen der Veranschaulichung der beschriebenen Prinzipien. Es handelt sich um frei erfundene Daten.

Codebeispiele erheben insbesondere bei JSON-Code nicht den Anspruch auf syntaktische Korrektheit und Vollständigkeit. Dementsprechend können in Codebeispielen Auslassungen vorkommen, die mit `...` gekennzeichnet werden.

[9] RFC2119 http://tools.ietf.org/html/rfc2119

1.5.4 Namespace-Präfixe für Objekt- und Datentypen

Bei der Erwähnung von Objekttypen, die in dieser Spezifikation beschrieben werden, wird in der Regel ein Präfix `oparl:` vor den Namen gesetzt, z. B. “oparl:Organization”. Damit soll verdeutlicht werden, dass der Objekttyp innerhalb der OParl-Spezifikation gemeint ist.

Das Präfix `oparl:` steht hierbei für die folgende Namespace-URL:

```
https://schema.oparl.org/1.1/
```

Dadurch kann eine Typenangabe wie `oparl:Organization` eindeutig in die folgende URL übersetzt werden:

```
https://schema.oparl.org/1.1/Organization
```

1.6 Datenschutz

Gemäß der Grundlage “öffentliche Daten nutzen, private Daten schützen” hat Datenschutz auch bei OParl eine hohe Priorität. Hierbei ist die deutsche Datenschutz-Gesetzgebung zu beachten.

Um personenbezogene Daten zu veröffentlichen, ist üblicherweise eine explizite Zustimmung der betroffenen Person erforderlich. Dies gilt für die bestehende Weboberfläche des Ratsinformationssystems ebenso wie für die OParl- Schnittstelle. Besonders zu beachten sind hierbei unter anderem E-Mail- Adressen, Anschriften, Fotos und Anwesenheitslisten. Es wird empfohlen, vor Veröffentlichung über die Schnittstelle den zuständigen Datenschutzbeauftragten zu kontaktieren.

1.7 OParl Governance

Im Verlauf der Weiterentwicklung können wie bei jedem Standardisierungsprozess Konflikte über die Ausrichtung und die Implementierung entstehen. Ist dies der Fall, so sollte als erstes der Issue Tracker auf Github für eine offene Diskussion und eine konstruktive Lösung verwendet werden.

Sollte es auf Github wider Erwarten keine Lösung geben, wird die Entscheidung an das OParl-Schlichtungsgremium weitergegeben. In diesem Gremium vertreten sind Entwickler, Anwender und Datenbereitsteller, so dass eine ausgewogene Weiterentwicklung im Interesse aller Akteure gewahrt bleibt.

Es ist natürlich unabhängig davon jederzeit erlaubt, einen Fork der OParl-Schnittstelle zu erstellen und dort neue zunächst nicht mehrheitsfähige Konzepte, Features und Funktionen auszuprobieren.

1.8 Autoren

1.8.1 Kernteam

Stefan Graupner, Ernesto Ruge, Konstantin Schütze

1.8.2 OParl 1.0

Folgende Personen haben an OParl 1.0 mitgewirkt:

Jayan Areekadan, Jan Erhardt, Lucas Jacob, Jens Klessmann (*), Andreas Kuckartz (**), Babett Schalitz, Tim Scheuermann, Christine Siegfried (*), Ralf Sternberg, Marian Steinbach (*), Bernd Thiem, Thomas Tursics, Jakob Voss, Marianne Wulff(*)

1.8.3 OParl 1.1

Folgende Personen haben an OParl 1.1 mitgewirkt:

grindhold, Simeon Maxein, Sami Mussbach, Ralf Sternberg

(*): Initiator(in), (**): bis 4.7.2014

2 Prinzipien und Funktionen der Schnittstelle

2.1 Designprinzipien

2.1.1 Aufbauen auf gängiger Praxis

Grundlage für die Erarbeitung der OParl-Spezifikation in der vorliegenden Version ist eine Analyse von aktuell (2012 bis 2016) in Deutschland etablierten parlamentarischen Informationssystemen und ihrer Nutzung. Erklärtes Ziel für diese erste Version ist es, mit möglichst geringem Entwicklungsaufwand auf Seite der Softwareanbieter und ebenso geringem Migrationsaufwand auf Seite der Betreiber zu einer Bereitstellung von parlamentarischen Informationen über eine OParl-API zu gelangen. Hierbei war es von entscheidender Bedeutung, dass sich die Informationsmodelle der einschlägigen Softwareprodukte stark ähneln. Für die OParl-Spezifikation wurde sozusagen ein Datenmodell als “gemeinsamer Nenner” auf Basis der gängigen Praxis konstruiert.

2.1.2 Verbesserung gegenüber dem Status Quo wo möglich

Dort, wo es dem Ziel der einfachen Implementierbarkeit und der einfachen Migration nicht im Weg steht, erlauben sich die Autoren dieser Spezifikation, auch Funktionen aufzunehmen, die noch nicht als gängige Praxis im Bereich der Ratsinformationssysteme bezeichnet werden können oder welche nur von einzelnen Systemen unterstützt werden. Solche Funktionen sind dann so integriert, dass sie nicht als zwingende Anforderung gelten.

Ein Beispiel für eine derartige Funktion ist die Abbildung von Geodaten im Kontext von Drucksachen (`oparl:Paper`), um beispielsweise die Lage eines Bauvorhabens, das in einer Beschlussvorlage behandelt wird, zu beschreiben. Zwar ist den Autoren nur ein einziges parlamentarisches Informationssystem[10] in Deutschland bekannt, das Geoinformationen – und zwar in Form von Punktdaten, also einer Kombination aus Längen- und Breitengradangaben – mit Dokumenten verknüpft. Der Vorteil dieser Funktion ist jedoch anhand zahlreicher Anwendungsszenarien, wie z.B. dem Bauinformationssystem “Bürger baut Stadt”[11], belegbar. Somit ist in der vorliegenden OParl-Spezifikation die Möglichkeit beschrieben, Geodaten-Objekte einzubetten.

[10] Das Ratsinformationssystem BoRis, eine Eigenentwicklung der Stadt Bonn http://www2.bonn.de/bo_ris/ris_sql/agm_index.asp

[11] bürgerbautstadt, http://www.buergerbautstadt.de

Die Angabe eines einzelnen Punktes ist dabei der einfachste Fall. Die Spezifikation erlaubt auch die Kodierung von mehreren Objekten, die Punkte, Linien oder Polygone repräsentieren können. Vgl. dazu `oparl:Location`.

Auch die Ausgabe einer Nur-Text-Version im Kontext der Datei (`oparl:File`), das den barrierefreien Zugriff auf Inhalte oder Indexierung für Volltextsuchfunktionen deutlich vereinfacht, ist eine Möglichkeit, die in der gängigen Praxis noch nicht zu finden ist. Ebenso die Möglichkeit, Beziehungen zwischen einzelnen Dateien herzustellen, um so z.B. von einer Datei zu anderen Dateien mit identischem Inhalt, aber in anderen technischen Formaten zu verweisen, etwa von einer ODT-Datei zu einer PDF-Version.

2.1.3 Selbstbeschreibungsfähigkeit

Ausgaben des Servers sollten so beschaffen sein, dass sie für menschliche Nutzerinnen weitgehend selbsterklärend sein können. Dies betrifft besonders die Benennung von Objekten und Objekteigenschaften.

Um den Kreis der Entwicklerinnen und Entwickler, die mit einer OParl-API arbeiten können, nicht unnötig einzuschränken, wird hierbei grundsätzlich und soweit sinnvoll auf englischsprachige Begrifflichkeiten gesetzt.

2.1.4 Erweiterbarkeit

Implementierer sollen in der Lage sein, über eine OParl-konforme Schnittstelle auch solche Informationen auszugeben, die nicht im Rahmen des OParl-Schemas abgebildet werden können. Dies bedeutet zum einen, dass ein System Objekttypen unterstützen und ausliefern darf, die nicht (oder noch nicht) im OParl-Schema beschrieben sind. Das bedeutet auch, dass Objekttypen so um eigene Eigenschaften erweitert werden können, die nicht im OParl Schema beschrieben sind.

Ein weiterer Aspekt betrifft die Abwärtskompatibilität, also die Kompatibilität von OParl-Clients mit zukünftigen Schnittstellen. So können beispielsweise zukünftige Erweiterungen des OParl-Schemas, etwa um neue Objekttypen, genauso durchgeführt werden, wie die Erweiterungen um herstellerspezifische Objekttypen. Ein Client muss diese Anteile nicht auswerten, sofern sie nicht für die Aufgabe des Clients relevant sind. Es bedeutet im Umkehrschluss allerdings auch, dass ein Client nicht fehlschlagen darf, falls derartige Erweiterungen vorhanden sind.

2.1.5 Browseability/Verlinkung

Klassische Webservice-Schnittstellen erfordern von den Entwicklern vollständige Kenntnis der angebotenen Einstiegspunkte und Zugriffsmethoden, gepaart mit sämtlichen unterstützten URL-Parametern, um den vollen Funktionsumfang der Schnittstelle ausschöpfen zu können.

Parlamentarische Informationen sind weitgehend in Form von Graphen aufgebaut. Das bedeutet, dass Objekte häufig mit einer Vielzahl anderer Objekte verknüpft sind. So ist eine Person beispielsweise Mitglied in mehreren Gremien, das Gremium hat mehrere Sitzungen abgehalten und zu diesen Sitzungen gibt es jeweils zahlreiche Drucksachen, die ihrerseits wieder zahlreiche Dokumente enthalten.

Eine OParl-Schnittstelle gibt jedem einzelnen Objekt eine eindeutige Adresse, eine URL. Somit kann die Schnittstelle den Verweis von einem Objekt, beispielsweise einem Gremium, auf ein anderes Objekt, etwa ein Mitglied des Gremiums, dadurch ausgeben, dass im Kontext des Gremiums die URL des Mitglieds ausgeben wird. Der Client kann somit ausgehend von einem bestimmten

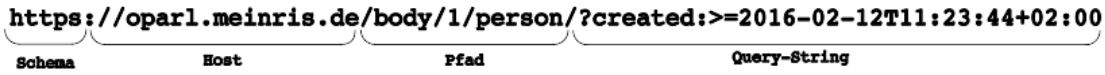

Abbildung 1: Aufbau einer URL

Objekt die zugehörigen Objekte im System finden, indem er einfach den angebotenen URLs folgt. Dieses Prinzip wird auch "Follow Your Nose"[12] genannt.

2.2 Zukunftssicherheit

Sollte in Zukunft eine zu OParl 1.0 inkompatible Version 2.0 erscheinen, kann ein Server beide Versionen gleichzeitig unterstützen, um mit OParl 1.0 Clients kompatibel zu bleiben. Dazu muss der Server die OParl 2.0-Schnittstelle unter einer eigenen URL parallel zur bestehenden OParl 1.0-Schnittstelle anbieten, siehe Kapitel System.

2.3 URLs

Den URLs (für *Uniform Resource Locators*) kommt eine besondere Bedeutung zu und es werden deshalb eine Reihe von Anforderungen an deren Aufbau und Eigenschaften gestellt. Die allgemeine Funktionsweise von URLs ist in RFC 3986 beschrieben[13].

Grundsätzlich **müssen** alle Zugriffe zustandslos erfolgen können, also ohne Sessioninformationen wie Cookies. Das bedeutet, dass alle Informationen, die zum Abrufen eines Objekts nötig sind, in der URL vorhanden sein müssen.

2.3.1 URL-Kanonisierung

Um Objekte eindeutig identifizieren zu können ist es notwendig, dass ein Server für ein Objekt genau eine unveränderliche URL benutzt. Diese Festlegung auf genaue eine eindeutige URL wird Kanonisierung genannt. Ein Server **muss** deshalb für jedes seiner Objekte eine kanonische URL bestimmen können.

Es wird empfohlen keine IP-Adressen in URLs zu benutzen, sondern einen mit Bedacht gewählten Hostnamen einzusetzen. Das ist vor allem im Hinblick auf die Langlebigkeit der URLs wichtig.

Um die Kanonisierung zu gewährleisten **sollten** OParl-Server so konfiguriert werden, dass sie nur über eine bestimmte Domain erreichbar sind. OParl-Server **sollten** dagegen möglichst **nicht** nur über eine IP-Addresse sowieso möglichst auch **nicht** über weitere, nicht kanonische URLs erreichbar sein.

Wenn ein Server auch durch eine nicht-kanonische URL erreichbar ist, dann **sollte** eine entsprechende HTTP-Anfrage mit einer Weiterleitung auf die entsprechende kanonische URL und HTTP-Status-Code 301 beantwortet werden. Zur Überprüfung kann z.B. der `Host`-Header einer HTTP-Anfrage verwendet werden.

Beim Pfad-Bestandteil der URL **müssen** Server-Implementierer darüber hinaus beachten, dass zur kanonischen Schreibweise auch die Groß- und Kleinschreibung, die Anzahl von Schrägstrichen als Pfad-Trennzeichen und die Anzahl von führenden Nullen vor numerischen URL-Bestandteilen gehört.

[12] http://patterns.dataincubator.org/book/follow-your-nose.html
[13] RFC 3986: http://tools.ietf.org/html/rfc3986

Die Kanonisierung umfasst auch den Query-String-Bestandteil der URL. Wie auch beim Pfad gilt, dass für jeden Parameter und jeden Wert im Query-String genau eine kanonische Schreibweise gelten **muss**.

Darüber hinaus **sollte** der Server-Implementierer darauf achten, Query-String-Parameter immer nach demselben Prinzip zu sortieren. Als Beispiel: Die beiden URLs

```
https://oparl.example.org/members?body=1&committee=2
https://oparl.example.org/members?committee=2&body=1
```

unterscheiden sich lediglich in der Reihenfolge der Query-String-Parameter. Da sie jedoch nicht identisch sind, könnten Clients annehmen, dass beide URLs verschiedene Objekte repräsentieren.

Clients **sollen** die vom Server gelieferten URLs bei Anzeige, Speicherung und Weiterverarbeitung nicht verändern.

2.3.2 HTTP und HTTPS

Der Einsatz des verschlüsselten HTTPS wird empfohlen. Bei Verwendung von HTTPS wird allen URLs "https://" voran gestellt, ansonsten beginnen URLs mit "http://".

Aus Gründen der URL-Kanonisierung ist es **zwingend** notwendig, dass ein Server-Betreiber sich entweder für HTTP oder für HTTPS entscheidet. Es jedoch möglich, eine Weiterleitung (HTTP Status-Code 301) einzurichten. Eine Weiterleitung von HTTPS auf HTTP wird **nicht empfohlen**.

2.3.3 Langlebigkeit

Weiterhin sollen URLs langlebig sein, sodass sie möglichst lange zur Abfrage des dazugehörigen Objekts verwendet werden können.

In URLs **sollten** deshalb nur Eigenschaften des Objekts aufgenommen werden, die nicht verändert werden. Ändert sich beispielsweise die Kennung einer Drucksache im Verlauf ihrer Existenz, dann scheidet sie für die Bildung der URL aus.

Des Weiteren sollen Eigenschaften der Implementierung nicht sichtbar sein. Ist ein OParl-Server beispielsweise in PHP geschrieben, **sollte** dies **nicht** dazu führen, dass im Pfad ein Bestandteil wie "oparl.php/" erscheint.

Weitere Empfehlungen für langlebige URLs liefern Tim Berners-Lee[14] sowie die Europäische Kommission[15].

2.4 JSON-Ausgabe

Ein OParl-Server **muss** Objekte in Form von JSON ausgeben. Die Abkürzung JSON steht für "JavaScript Object Notation". Das JSON-Format ist in RFC 7159[16] beschrieben.

[14] Berners-Lee, Tim: Cool URIs don't change. http://www.w3.org/Provider/Style/URI.html

[15] Study on persistent URIs, with identification of best practices and recommendations on the topic for the MSs and the EC. (PDF) https://joinup.ec.europa.eu/sites/default/files/D7.1.3%20-%20Study%20on%20persistent%20URIs.pdf

[16] RFC 7159: https://tools.ietf.org/html/rfc7159

Sämtliche JSON-Ausgabe **muss** in UTF-8 ohne Byte Order Mark (BOM) geschehen. Dies entspricht RFC 7159 Section 8.1[17]. Gemäß RFC 7159 Section 7[18] **darf** UTF-8 String-Escaping verwendet werden. XML-/HTML-String-Escaping **darf nicht** verwendet werden.

Eine Syntaxübersicht und weitere Implementierungshinweise finden sich auf json.org.

Es ist gestattet, weitere zur JSON-Ausgabe semantisch identische Formate[19] anzubieten. Da diese jedoch nicht Bestandteil der Spezifikation sind, **sollten** sich Clients nicht auf deren Vorhandensein verlassen.

2.4.1 In OParl verwendete Datentypen

In OParl werden alle in JSON definierten Dateitypen verwendet:

object: Objects entsprechen der Definition des Objects in RFC 7159 Section 4
array: Arrays entsprechen der Definition des Arrays in RFC 7159 Section 5
integer: Integers entsprechen der Definition des Integer-Parts der Number aus RFC 7159 Section 6
boolean: Booleans entsprechen der Definition von Boolean in RFC 7159 Section 3
string: Strings entsprechen der Definition der Unicode-Strings aus RFC 7159 Section 7

In OParl werden verschiedene String-Typen verwendet. Wenn von diesen Typen gesprochen wird, so wird automatisch ein JSON-String vorausgesetzt:

url: Eine URL ist ein String, der entsprechend des URL-Kapitels formatiert wurde.
url (Object): Eine URL mit in Klammern angehängtem Objektname beschreibt eine URL auf eben diesen Objekttypus.
date: Entspricht einem Datum ohne Uhrzeit und ohne Zeitzone, wie sie im folgenden Abschnitt beschrieben werden.
date-time: Entspricht einem Datum und einer Uhrzeit mit Zeitzone, wie sie im folgenden Abschnitt beschrieben werden.

2.4.2 Datums- und Zeitangaben

Für Datums- und Zeitangaben wird eine Spezielisierung der in ISO 8601 beschriebenen Formate verwendet. Ein Datum (date) **muss** muss die Form `yyyy-mm-dd` besitzen und ein Zeitpunkt (date-time) **muss** in der Form `yyyy-mm-ddThh:mm:ss±hh:mm` angegeben werden.

Beispiel für ein Datum: `1969-07-21`

Beispiel für einen Zeitpunkt: `1969-07-21T02:56:00+00:00`

2.4.3 `null`-Werte und leere Listen

JSON erlaubt es grundsätzlich, Eigenschaften mit dem Wert `null` zu versehen. Eigenschaften **sollten** nicht mit dem Wert `null` ausgegeben werden, wenn zu einer Eigenschaft keine Daten vorliegen. Obligatorische Eigenschaften **dürfen nicht** den Wert `null` haben.

[17] RFC 7159 Section 8.1
[18] RFC 7159 Section 7
[19] Zu semantisch identischen Formaten zählen u.a.: YAML, MessagePack, etc.

Im Fall von Arrays erlaubt JSON grundsätzlich die Ausgabe von `[]` für leere Arrays. Wie bei `null` wird auch hier **empfohlen**, auf die Ausgabe einer Eigenschaft mit dem Wert `[]` zu verzichten, wenn zu einer Eigenschaft keine Daten vorliegen. Bei obligatorischen Eigenschaften **muss** jedoch eine leere Liste ausgegeben werden.

Bei nicht obligatorischen Eigenschaften sollte gleichermaßen auf die Ausgabe eines leeren Strings verzichtet werden.

2.5 Objektlisten und Paginierung

Oft wird für ein Attribut kein Wert ausgegeben, sondern ein anderes Objekt oder eine Liste von Objekten. Dabei kann eine Referenz auf das Objekt bzw. die Objektliste angegeben werden, oder das Objekt bzw. die Objektlist wird intern ausgegeben. Beide Verfahren sollen im Folgenden erklärt werden. Zu beachten ist, dass für jedes Listenattribut festgelegt ist, welches dieser Verfahren jeweils zu verwenden ist. Diese Information ist den Schemadefinitionen zu entnehmen.

2.5.1 Referenzierung von Objekten via URL

Bei der Referenzierung einzelner Objekte wird eine URL angegeben, welche auf das entsprechende Objekt verweist. Der Typ ist hierbei ein `string (url: Objekt-ID)`. Ein Beispiel hierfür ist `subOrganizationOf` in `Organization`:

```
{
  "id": "https://oparl.example.org/organization/1",
  "type": "https://schema.oparl.org/1.1/Organization",
  "subOrganizationOf": "https://oparl.example.org/organization/2"
  ...
}
```

Es kann auch eine Liste von Referenzen ausgegeben werden. Der Typ ist in diese Fall `array of string (url: Objekt-ID)`.

Ein Beispiel hierfür ist `meeting` in `Organization`:

```
{
  "id": "https://oparl.example.org/organization/1",
  "type": "https://schema.oparl.org/1.1/Organization",
  "meeting": [
    "https://oparl.example.org/meeting/1",
    "https://oparl.example.org/meeting/2",
    "https://oparl.example.org/meeting/3",
  ]
  ...
}
```

2.5.2 Interne Ausgabe von Objekten

Objekte können auch intern ausgegeben werden. Dabei wird das gesamte Objekt als Wert eines Attributs angegeben. Ein Beispiel für ein internes Objekt ist `location` in `oparl:Body`:

```
{
  "id": "https://oparl.example.org/body/1",
  "type": "https://schema.oparl.org/1.1/Body",
  "location": {
    "id": "https://oparl.example.org/location/1",
    "type": "https://schema.oparl.org/1.1/Location",
    "description": "Ratshausvorplatz 1, 12345 Beispielstadt"
  },
  ...
}
```

Ebenso kann eine Liste von Objekten intern ausgegeben werden. Hier das Beispiel des Attributes `membership` in `oparl:Person`.

```
{
  "id": "https://oparl.example.org/person/1",
  "type": "https://schema.oparl.org/1.1/Person",
  "membership": [
    {
      "id": "https://oparl.example.org/memberships/385",
      "organization": "https://oparl.example.org/organizations/5",
      "role": "Vorsitzende",
      "votingRight": true,
      "startDate": "2013-12-03"
    },
    {
      "id": "https://oparl.example.org/memberships/693",
      "organization": "https://oparl.example.org/organizations/9",
      "role": "Sachkundige Bürgerin",
      "votingRight": false,
      "startDate": "2013-12-03",
      "endDate": "2014-07-28"
    }
  ],
  ...
}
```

Bei der internen Ausgabe von Objekten **darf** der Server keine gelöschten Objekte ausgeben.

2.5.3 Externe Objektlisten

Es können auch Referenzen zu sogenannten externen Objektlisten angegeben werden. Die externe Liste enthält dann die betreffenden Objekte in Form einer Listenausgabe. Ein Beispiel dafür ist `organization` in `oparl:Body`.

`oparl:Body`:

```
{
  "id": "https://oparl.example.org/body/1",
  "type": "https://schema.oparl.org/1.1/Body",
  "organization": "https://oparl.example.org/body/1/organization"
```

```
  ...
}
```

Die externe Objektliste:

```
{
    "data": [
      {
        "id": "https://oparl.example.org/organization/1",
        "type": "https://schema.oparl.org/1.1/Organization",
        "name": "Organisation Nummer 1",
        ...
      },
      {
        "id": "https://oparl.example.org/organization/2",
        "type": "https://schema.oparl.org/1.1/Organization",
        "name": "Organisation Nummer 2",
        ...
      },
      {
        "id": "https://oparl.example.org/organization/3",
        "type": "https://schema.oparl.org/1.1/Organization",
        "name": "Organisation Nummer 3",
        ...
      },
    ],
    ...
}
```

2.5.4 Paginierung

Für externe Objektlisten ist eine Aufteilung sogenannte *Listenseiten* vorgesehen, wobei jede Listenseite eine eigene URL erhält. Das dient dazu, die bei der jeweiligen Anfrage übertragenen Datenmengen und Antwortzeiten zu begrenzen.

Die Entscheidung, ob eine externe Objektliste mit Paginierung ausgegeben wird, liegt allein beim Server. Bei Listen mit mehr als 100 Einträgen wird dies **empfohlen**.

Ein Server **muss** für eine stabile Sortierung von Listeneinträgen sorgen. Das heißt, dass die Sortierung der Einträge einem konstanten Prinzip folgt und sich nicht von Abfrage zu Abfrage ändert. Das kann z.B. durch die Sortierung von Objekten nach einer eindeutigen und unveränderlichen ID erreicht werden.

Jede Listenseite **muss** die Attribute folgenden Attribute enthalten:

- **data** (Array der intern ausgegebenen Objekte)
- **pagination** (Object)
- **links** (Object)

Für `pagination` sind die folgenden Attribute festgelegt, die alle **optional** sind:

- `totalElements`: Gibt die Gesamtanzahl der Objekte in der Liste an. Diese Zahl kann sich unter Umständen bis zum Aufruf der nächsten Listenseiten ändern.
- `elementsPerPage`: Gibt die Anzahl der Objekte pro Listenseite an. Dieser Wert muss auf allen Listenseiten bis auf die letzte gleich sein.
- `currentPage`: Gibt die aktuelle Seitenzahl in der Liste an.
- `totalPages`: Gibt die Gesamtanzahl der Seiten in der Liste an.

Für `links` sind folgende Attribute festgelegt, die bis auf `next` alle **optional** sind:

- `first`: URL der ersten Listenseite
- `prev`: URL der vorherigen Listenseite
- `self`: Die kanonische URL dieser Listenseite
- `next`: URL der nächsten Listen. Für alle Seiten bis auf die letzte ist die Angabe dieser URL **zwingend**.
- `last`: URL der letzten Listenseite
- `web`: s. web. Neu in OParl 1.1

```
{
    "data": [
        {...},
        {...},
        ...
    ],
    "pagination": {
        "totalElements": 50000,
        "elementsPerPage": 100,
        "currentPage": 3,
        "totalPages":500
    },
    "links": {
        "first": "https://oparl.example.org/organization/",
        "prev": "https://oparl.example.org/organization/?page=2",
        "self": "https://oparl.example.org/organization/?page=3",
        "next": "https://oparl.example.org/organization/?page=4",
        "last": "https://oparl.example.org/organization/?page=500",
        "web": "https://web.example.org/organization/?page=500"
    }
}
```

2.5.5 Filter

Externe Objektlisten können mit den URL-Parametern `created_since`, `created_until`, `modified_since` und `modified_until` eingeschränkt werden. Diese Parameter beziehen sich auf die entsprechenden Attribute der jeweiligen Objekte, wobei reservierte Zeichen URL-Kodiert werden müssen. Ein Server muss diese Parameter bei allen externen Objektlisten unterstützen.

Neu in OParl 1.1: Wenn ein Client den Parameter `omit_internal` mit dem Wert `true` angibt, dann **soll** der Server auf die Ausgabe von internen Listen verzichten. Konkret bedeutet das, dass die folgenden Attribute nicht ausgegeben werden müssen:

- `auxiliaryFile` in `AgendaItem`
- `auxiliaryFile` in `Meeting`
- `auxiliaryFile` in `Paper`
- `location` in `Paper`
- `membership` in `Person`
- `agendaItem` in `Meeting`
- `legislativeTerm` in `Body`

Die Filter werden vom Client benutzt, indem die gewünschten URL-Parameter an die URL der ersten Listenseite angehängt werden. Bei allen weiteren Seiten, genauer gesagt bei den Werten von `links`, **muss** der Server sicherzustellen, dass die verwendeten Filter erhalten bleiben.

Neu in OParl 1.1: Ein Server **muss** für den im nächsten Abschnitt beschrieben Aktualisierungsmechanismus auch die den Filtern entsprechenden gelöschten Objekte ausgeben, wenn der Parameter `modified_since` gesetzt ist (s. OParl 1.1). Wenn `modified_since` nicht gesetzt ist, dann **dürfen** die gelöschten Objekte **nicht** ausgegeben werden. Dadurch kann sich ein Client effizient darüber informieren, welche der Objekte in seinem lokalen Bestand gelöscht wurden.

Lautet die URL für eine Liste von Drucksachen wie folgt:

```
https://oparl.example.org/papers/
```

kann der Client die folgende URL bilden, um die Ausgabe der Liste auf Drucksachen einzuschränken, die seit dem 1. Januar 2014 veröffentlicht wurden:

```
https://oparl.example.org/papers/?created_since=2014-01-01T00%3A00%3A00%2B01%3A00
```

Mehrere Parameter können auch gemeinsam verwendet werden. So kann man z.B. eine Einschränkung vom 1.1.2014 bis zum 31.1.2014 vornehmen:

```
https://oparl.example.org/papers/?created_since=2014-01-01T00%3A00%3A00%2B01%3A00&created_until=2
```

Die genannten URL-Parameter erwarten grundsätzlich eine vollständige `date-time`-Angabe.

Des Weiteren kann ein Client die Anzahl der Objekte pro Listenseite durch den URL-Parameter `limit` begrenzen, der sich auf das gleichnamige Attribut bezieht. Ein Client **darf nicht** erwarten, dass sich ein Server an seine `limit`-Anfrage hält.

2.5.6 Der Aktualisierungsmechanismus

Dieser Abschnitt ist neu in OParl 1.1.

Der Hauptnutzen der Filter ist die Möglichkeit, einen lokalen Datenbestand inkrementell zu aktualisieren.

Ein Client könnte z.B. am 1.1.2014 um 2:00 Uhr deutscher Zeit die Liste aller Drucksachen herunterladen und in einer Datenbank speichern.

```
https://oparl.example.org/papers/
```

Um den Datenbestand am nächsten Tag zu aktualisieren, ruft der Client dieselbe URL auf, diesmal jedoch mit dem Parameter `modified_since` mit dem Wert `2014-01-01T02:00:00+01:00` und mit `omit_internal`.

```
https://oparl.example.org/papers/?modified_since=2014-01-01T02%3A00%3A00%2B01%3A00&omit_internal=
```

Diese Liste ist in der Regel deutlich kürzer als die Liste aller Objekte, sodass die Aktualisierung bedeutend schneller ist als der erste Abruf. Der Client muss außerdem nur noch eine deutlich kleinere Menge an Objekten in die Datenbank einfügen, aktualisieren oder löschen, um den gleichen Datenstand wie der Server zu haben.

2.6 Cross-Origin Resource Sharing (CORS)

Wenn Webbrowser mittels Skript auf JSON-Ressourcen zugreifen sollen unterliegen diese Zugriffe üblicherweise einer *Same-Origin-Policy* (SOP). Das heißt, eine Anfrage ist nur an den Server zulässig, der auch das initiierende Skript ausgeliefert hat. Anfragen an andere Server werden vom Browser blockiert. Diese Einschränkung dient im Allgemeinen der Sicherheit von Webbrowsern.[20]

Um die Daten von OParl-Servern auch im Kontext von Webanwendungen flexibel nutzen zu können, ist die Überwindung der SOP nötig. Hierzu dient *Cross-Origin Resource Sharing* (CORS)[21]. Mittels CORS kann ein Server mitteilen, dass bestimmte von ihm ausgelieferte Ressourcen auch innerhalb von Webapplikationen genutzt werden dürfen, die nicht vom selben Server ausgeliefert werden. Technisch wird dies durch Ausgabe zusätzlicher HTTP-Header erreicht.

OParl-Server **müssen** für jegliche Anfrage, die mit der Ausgabe von JSON-Daten beantwortet wird (das sind alle Anfragen außer Dateizugriffe) den folgenden HTTP-Antwort-Header senden:

```
Access-Control-Allow-Origin: *
```

Der HTTP-Antwort-Header `Access-Control-Allow-Methods` **sollte** darüber hinaus **nicht** gesetzt sein, oder **muss** die Methode `GET` beinhalten.

Entwicklerinnen von Webanwendungen sollten sich darüber bewusst sein, dass durch die direkte Einbindung von Skripten Dritter in ihre Anwendungen mögliche Sicherheitsrisiken entstehen. Für den Fall, dass ein OParl-Server, etwa in Folge einer Manipulation, Schadcode ausliefert, könnte dieser unmittelbar von Skripten im Browser ausgeführt werden.

2.7 Dateizugriffe

Mit dem Begriff "Datei" sind im Sinne dieser Spezifikation alle Ressourcen gemeint, die von einem OParl-Server zur Verfügung gestellt werden und deren Metadaten über die JSON-API als `oparl:File` abgerufen werden können. Es handelt sich dabei beispielsweise um Textdokumente im PDF-Format oder Abbildungen im JPEG- oder PNG-Format.

Jede Datei **muss** dabei mit einer HTTP-GET-Anfrage abrufbar sein.

[20] vgl. Wikipedia: Same-Origin-Policy https://de.wikipedia.org/wiki/Same-Origin-Policy
[21] Cross Origin Resource Sharing - W3C Recommendation 16. Januar 2014: http://www.w3.org/TR/cors/

2.7.1 Empfehlungen für Dateizugriffe

- Ein Server **sollte** die Verwendung von Kompression gemäß dem HTTP-Standard unterstützen.
- Ein Server **sollte** "Conditional GET", insbesondere `If-Modified-Since` und `If-None-Match` sowie "Chunked GET" unterstützen.
- Die Ausgabe der HTTP-Header `Last-Modified`, `Content-Length` und `ETag` ist **empfohlen**.
- Bei gelöschten Dateien **sollte** der HTTP-Statuscode `410` verwendet werden.

2.7.2 Allgemeiner Zugriff und expliziter Download

Mit der im `oparl:File` **zwingend** anzugebenden Eigenschaft `accessUrl` liefert der Server dem Client eine URL, die dem allgemeinen Zugriff auf die Datei dient. Beim Zugriff auf dieser URL **darf** der Server **nicht** den `Content-Disposition`-Header mit dem Parameter `attachment` senden. [22]

Es wird daher **empfohlen**, zusätzlich eine Eigenschaft `downloadUrl` anzubieten. Beim Zugriff auf die Download-URL **muss** der Server in der HTTP-Antwort einen `Content-Disposition`-Header senden, der als ersten Parameter den Typ `attachment` enthält und mit dem `filename`-Parameter den Namen der Datei angibt.

Beispiel:

```
Content-Disposition: attachment; filename="2014-08-22 Rat Wortprotokoll.pdf"
```

2.8 Gelöschte Objekte

In OParl **dürfen** Objekte **nicht** einfach gelöscht werden, sodass unter der betreffenden URL kein gültiges Objekt ausgeliefert wird. Stattdessen wird ein sogenanntes *soft delete* verwendet.

Hintergrund ist, dass OParl-Clients bei der Aktualisierung ihres Datenbestandes, z.B. mit den Filtern `modified_since` bzw. `created_since`, erfahren können müssen, welche Objekte gelöscht wurden.

Dies wird durch die folgenden Regeln gewährleistet.

Wenn ein Objekt gelöscht wird,

- **muss** das Objekt das zusätzliche Attribut `deleted` mit dem Wert `true` bekommen
- **muss** das Attribut `modified` auf den Zeitpunkt der Löschung setzen
- **müssen** die Attribute `id`, `type` und `created` erhalten bleiben
- **dürfen** alle weiteren Attribute entfernt werden

Als HTTP-Statuscode **muss** weiterhin 200 verwendet werden.

Neu in OParl 1.1: Die Objekte *LegislativeTerm*, *Membership*, *AgendaItem* und *Consultation* dürfen nicht mehr einfach gelöscht werden. Um Kompatibilität zu OParl 1.0 zu gewährleisten muss weiterhin der Wert `modified` aller Objekte aktualisiert werden, in die dieses Objekt eingebettet war.

[22] vgl. RFC2138 http://www.ietf.org/rfc/rfc2183

2.8.1 Depublizierung von Objekten

Da es sich bei OParl um eine Schnittstelle für öffentliche Daten handelt werden depublizierte Objekte im Sinne der Schnittstelle gelöscht und **sollen** wie oben behandelt werden.

2.9 Ausnahmebehandlung

Wenn ein Server eine Anfrage nicht bearbeiten kann, z.B. weil die URL ungültig ist oder das angefragte Objekt nicht existiert, dann **sollte** er mit dem entsprechenden HTTP-Statuscode antworten.

Ein Server **sollte** in diesem Fall ein Objekt ausgeben, das die folgenden 3 Attribute enthält:

- `type`: Enthält als Wert die URL `https://schema.oparl.org/1.1/Error`
- `message`: Eine Fehlermeldung, die zur Anzeige für einen Nutzer gedacht ist. Die Fehlermeldung sollte deshalb in der Sprache der durch die Schnittstelle ausgelieferten Inhalte verfasst sein
- `debug`: Zusätzliche Informationen über den Fehler

Wenn ein Server ein solches Objekt ausgibt, dann **muss** er dazu einen HTTP-Statuscode senden, der einen Fehler anzeigt.

Ein Client **darf nicht** voraussetzen, dass er im Fall eines Fehlers verwertbare Informationen wie das oben beschriebene Fehlerobjekt erhält.

2.10 OParl Endpunkt

Als OParl Endpunkt bzw. Einsprungspunkt zur Schnittstelle wird ein `OParl:System` Objekt genutzt. Falls auf einem HTTP-Host mehrere OParl-Schnittstellen oder mehrere OParl Versionen parallel installiert sind, **müssen** diese eindeutige und voneinander unabhängige OParl-Endpunkte anbieten. Es ist allerdings möglich, eine Liste von `OParl:System`-Objekten auszugeben, die z.B. auf verschiedene OParl-Versionen einer Schnittstelle verweisen.

3 Schema

Dieses Kapitel beschreibt das Schema von OParl. Das Schema definiert die Objekttypen und ihre Eigenschaften. Darüber hinaus ist im Schema auch festgelegt, in welcher Beziehung verschiedene Objekttypen zu einander stehen.

3.1 Die Objekte

OParl nutzt die folgenden Objekte:

- `oparl:System`
- `oparl:Body`
- `oparl:LegislativeTerm`
- `oparl:Organization`
- `oparl:Person`

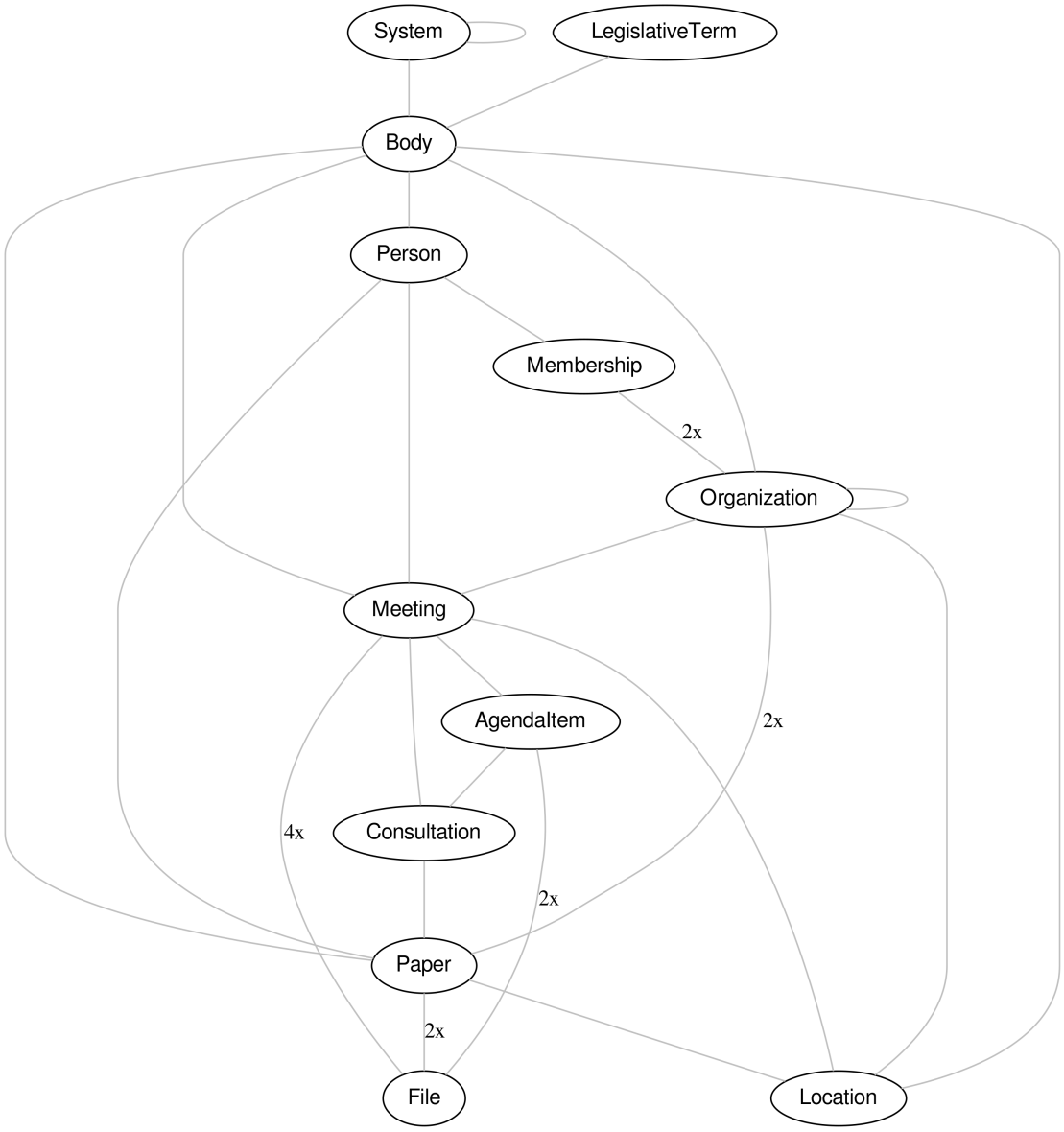

Abbildung 2: OParl Objekttypen: Ein Überblick. Die Zahl an den Verbindungslinien entspricht der Anzahl der Attribute, die eine oder mehrere Verknüpfungen herstellen.

- `oparl:Membership`
- `oparl:Meeting`
- `oparl:AgendaItem`
- `oparl:Paper`
- `oparl:Consultation`
- `oparl:File`
- `oparl:Location`

Einige Objekte werden intern in anderen Objekten ausgegeben:

- `oparl:LegislativeTerm` wird intern in `oparl:Body` ausgegeben
- `oparl:Membership` wird intern in `oparl:Person` ausgegeben
- `oparl:AgendaItem` wird intern in `oparl:Meeting` ausgegeben
- `oparl:Consultation` wird intern in `oparl:Paper` ausgegeben
- `oparl:File` wird intern in `oparl:Meeting`, `oparl:AgendaItem` und `oparl:Paper` ausgegeben
- `oparl:Location` wird intern in `oparl:Body`, `oparl:Organization`, `oparl:Meeting` und `oparl:Paper` ausgegeben

Grundsätzlich muss jedes Objekt unter seiner ID abrufbar sein - auch dann, wenn das Objekt in anderen Objekten intern ausgegeben wird. Bei der internen Ausgabe wird beim internen Objekt auf die Rückreferenz auf das Elternobjekt verzichtet.

Als Beispiel hier eine Ausgabe von `oparl:Meeting`, in welchem ein `oparl:File` enthalten ist:

```
{
    "id": "https://oparl.example.org/meeting/281",
    "type": "https://schema.oparl.org/1.1/Meeting",
    "name": "4. Sitzung des Finanzausschusses",
    "start": "2013-01-04T08:00:00+01:00",
    "end": "2013-01-04T12:00:00+01:00",
    "invitation": {
        "id": "https://oparl.example.org/files/57739",
        "name": "Einladung",
        "fileName": "einladung.pdf",
        "mimeType": "application/pdf",
        "date": "2012-01-08",
        "modified": "2012-01-08T14:05:27+01:00",
        "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
        "size": 82930,
        "accessUrl": "https://oparl.example.org/files/57739.pdf",
        "downloadUrl": "https://oparl.example.org/files/download/57739.pdf"
    }
    [...]
}
```

Das enthaltene `oparl:File` muss auch einzeln abgerufen werden können. Dabei kommt dann das Eltern-Objekt als zusätzliches Attribut hinzu.:

```
{
    "id": "https://oparl.example.org/files/57739",
```

```
    "type": "https://schema.oparl.org/1.1/File",
    "name": "Einladung",
    "fileName": "einladung.pdf",
    "mimeType": "application/pdf",
    "date": "2012-01-08",
    "modified": "2012-01-08T14:05:27+01:00",
    "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
    "size": 82930,
    "accessUrl": "https://oparl.example.org/files/57739.pdf",
    "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
    "meeting": [
        "https://oparl.example.org/meeting/281"
    ]
}
```

Das zusätzliche Attribut ist ein Array, da es auch möglich ist, dass Dateien von mehreren Hauptobjekten aus genutzt werden. Das kann z.B. bei `oparl:Location` vorkommen:

```
{
    "id": "https://oparl.example.org/locations/29856",
    "type": "https://schema.oparl.org/1.1/File",
    "description": "Honschaftsstraße 312, Köln",
    "geojson": {
        "type": "Feature",
        "geometry": {
            "type": "Point",
            "coordinates": [
                7.03291,
                50.98249
            ]
        }
    },
    "meeting": [
        "https://oparl.example.org/meeting/281",
        "https://oparl.example.org/meeting/766",
        "https://oparl.example.org/meeting/1002"
    ],
    "paper": [
        "https://oparl.example.org/paper/749",
        "https://oparl.example.org/paper/861",
        "https://oparl.example.org/paper/1077"
    ]
}
```

3.2 Übergreifende Aspekte

3.2.1 Vollständigkeit

Alle regulär öffentlich abrufbaren Informationen **sollten** auch in OParl ausgegeben werden, solange dies nicht den Datenschutzbestimmungen widerspricht. Daher sind sämtliche Felder im Schema als **empfohlen** zu behandeln, wenn nicht explizit etwas anderes angegeben wurde.

3.2.2 Herstellerspezifische Erweiterungen

In OParl können zusätzliche, herstellerspezifische Eigenschaften hinzugefügt werden. Dazu wird diesen Eigenschaften ein Herstellerprefix vorangestellt. So könnte man z.B. `oparl:Person` um eine Faxnummer erweitern:

```
"BeispielHersteller:faxNumber": "012345678",
```

3.2.3 URL-Pfade in den Beispielen

OParl-Clients wissen nichts vom Aufbau von Pfaden innerhalb von URLs, müssen dies nicht wissen, und es gibt deshalb in der OParl-Spezifikation keine Festlegungen dazu. Die in den Beispielen verwendeten URLs zeigen einen möglichen Weg zur Umsetzungen der Empfehlungen in URLs.

3.3 Eigenschaften mit Verwendung in mehreren Objekttypen

3.3.1 `id`

Die Eigenschaft `id` enthält den eindeutigen Bezeichner des Objekts, nämlich seine URL. Dies ist ein **zwingendes** Merkmal für jedes Objekt.

3.3.2 `type`

Objekttypenangabe des Objekts, **zwingend** für jedes Objekt. Der Wert ist eine Namespace-URL. Für die OParl-Objekttypen sind die folgenden URLs definiert:

Typ (kurz)	Namespace-URL
`oparl:AgendaItem`	https://schema.oparl.org/1.1/AgendaItem
`oparl:Body`	https://schema.oparl.org/1.1/Body
`oparl:Consultation`	https://schema.oparl.org/1.1/Consultation
`oparl:File`	https://schema.oparl.org/1.1/File
`oparl:LegislativeTerm`	https://schema.oparl.org/1.1/LegislativeTerm
`oparl:Location`	https://schema.oparl.org/1.1/Location
`oparl:Meeting`	https://schema.oparl.org/1.1/Meeting
`oparl:Membership`	https://schema.oparl.org/1.1/Membership
`oparl:Organization`	https://schema.oparl.org/1.1/Organization
`oparl:Paper`	https://schema.oparl.org/1.1/Paper
`oparl:Person`	https://schema.oparl.org/1.1/Person
`oparl:System`	https://schema.oparl.org/1.1/System

3.3.3 `name` und `shortName`

Beide Eigenschaften können bei vielen Objekttypen genutzt werden um den Namen des Objekts anzugeben. Üblicherweise ist `name` eine Pflichteigenschaft für den ausgeschriebenen offiziellen Namen, während `shortName` optional angegeben werden kann. Dies ist dann zu empfehlen, wenn zu einem Namen eine kurze bzw. kompakte und eine längere, aber weniger nutzerfreundliche Variante existieren. So ist “Innenministerium” die Kurzform des offiziellen “Bundesministerium

des Inneren".

3.3.4 `license`

Mit `license` wird angegeben, unter welcher Lizenz die Daten des jeweiligen Objekts stehen. [23]

Wird `license` im `oparl:System`-Objekt oder am `oparl:Body`-Objekt verwendet, dann bedeutet das, dass alle Objekte dieses Systems bzw. der Körperschaft unter der angegebenen Lizenz veröffentlicht werden, sofern nicht das einzelne Objekt eine anders lautende Lizenz-URL angibt. Es wird **empfohlen**, die Lizenzinformation sofern möglich global am `oparl:System` Objekt mitzuteilen und auf redundante Informationen zu verzichten.

3.3.5 `created`

Datum und Uhrzeit der Erstellung des jeweiligen Objekts.

Diese Eigenschaft **muss** in allen Objekttypen angegeben werden. Neu in OParl 1.1: Diese Eigenschaft muss auch in Objekten ausgegeben werden, die intern ausgegeben werden.

3.3.6 `modified`

Diese Eigenschaft kennzeichnet stets Datum und Uhrzeit der letzten Änderung des jeweiligen Objekts.

Diese Eigenschaft **muss** - genau wie `created` - in allen Objekttypen angegeben werden. Neu in OParl 1.1: Diese Eigenschaft muss auch in Objekten ausgegeben werden, die intern ausgegeben werden.

Es ist **zwingend**, dass bei jeder Änderung eines Objekts der Wert dieses Attributs auf die zu diesem Zeitpunkt aktuelle Uhrzeit gesetzt wird, da ein Client in der Regel seinen Datenbestand nur auf Basis dieses Attributs verlustfrei aktualisieren kann.

3.3.7 `keyword`

Die Eigenschaft `keyword` dient der optionalen Kategorisierung eines Objekts.

3.3.8 `web`

Gibt die URL einer Website an, die das Objekt im Browser darstellt. Das ist z.B. die HTML-Ansicht eines parlamentarischen Informationssystems.

3.3.9 `deleted`

Falls das Objekt gelöscht wurde, muss dieses gemäß Kapitel 2.8 das Attribut `deleted: true` bekommen.

[23] Verzeichnisse für Lizenz-URLs sind unter anderem unter http://licenses.opendefinition.org/ und https://github.com/fraunhoferfokus/ogd-metadata/blob/master/lizenzen/deutschland.json zu finden. Allgemeine Informationen zur Lizensierung von Open Data finden sich auch im Open Data Handbook der Open Knowledge Foundation unter http://opendatahandbook.org/de/how-to-open-up-data/apply-an-open-license.html.

3.4 System

Ein `oparl:System`-Objekt repräsentiert eine OParl-Schnittstelle für eine bestimmte OParl-Version. Es ist außerdem der Startpunkt für Clients beim Zugriff auf einen Server.

Möchte ein Server mehrere zueinander inkompatible OParl-Versionen unterstützen, dann **muss** der Server für jede Version eine eigenen OParl-Schnittstelle mit einem eigenen `System`-Objekt ausgeben.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`oparlVersion`	string	**ZWINGEND** Die URL der OParl-Spezifikation, die von diesem Server unterstützt wird. Aktuell kommt hier nur ein Wert in Frage. Mit zukünftigen OParl-Versionen kommen weitere mögliche URLs hinzu. Wert: `https://schema.oparl.org/1.1/`
`otherOparlVersions`	array of url (System)	Dient der Angabe von System-Objekten mit anderen OParl-Versionen.
`license`	url	Lizenz, unter der durch diese API abrufbaren Daten stehen, sofern nicht am einzelnen Objekt anders angegeben. Siehe `license`.
`body`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Körperschaften, die auf dem System existieren.
`name`	string	Nutzerfreundlicher Name für das System, mit dessen Hilfe Nutzerinnen und Nutzer das System erkennen und von anderen unterscheiden können.
`contactEmail`	string	E-Mail-Adresse für Anfragen zur OParl-API. Die Angabe einer E-Mail-Adresse dient sowohl NutzerInnen wie auch Entwicklerinnen von Clients zur Kontaktaufnahme mit dem Betreiber.
`contactName`	string	Name der Ansprechpartnerin bzw. des Ansprechpartners oder der Abteilung, die über die in `contactEmail` angegebene Adresse erreicht werden kann.
`website`	url	URL der Website des parlamentarischen Informationssystems
`vendor`	url	URL der Website des Softwareanbieters, von dem die OParl-Server-Software stammt.
`product`	url	URL zu Informationen über die auf dem System genutzte OParl-Server-Software
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/",
    "type": "https://schema.oparl.org/1.1/System",
    "oparlVersion": "https://schema.oparl.org/1.1/",
    "body": "https://oparl.example.org/bodies",
    "name": "Beispiel-System",
    "contactEmail": "info@example.org",
    "contactName": "Allgemeiner OParl Kontakt",
    "website": "http://www.example.org/",
    "vendor": "http://example-software.com/",
    "product": "http://example-software.com/oparl-server/",
    "otherOparlVersions": [
        "https://oparl2.example.org/"
    ],
    "created": "2011-11-11T11:11:00+01:00",
    "modified": "2012-11-11T11:11:00+01:00"
}
```

3.5 Body

Der Objekttyp oparl:Body dient dazu, eine Körperschaft zu repräsentieren. Eine Körperschaft ist in den meisten Fällen eine Gemeinde, eine Stadt oder ein Landkreis. In der Regel sind auf einem OParl-Server Daten von genau einer Körperschaft gespeichert und es wird daher auch nur ein Body-Objekt ausgegeben. Sind auf dem Server jedoch Daten von mehreren Körperschaften gespeichert, **muss** für jede Körperschaft ein eigenes Body-Objekt ausgegeben werden.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`system`	url (System)	System, zu dem dieses Objekt gehört.
`shortName`	string	Kurzer Name der Körperschaft.
`name`	string	**ZWINGEND** Der offizielle lange Name der Körperschaft.
`website`	url	Allgemeine Website der Körperschaft.
`license`	url	Lizenz, unter der die Daten dieser Körperschaft stehen, sofern nicht am einzelnen Objekt anders angegeben. Siehe `license`.
`licenseValidSince`	date-time	Zeitpunkt, seit dem die unter `license` angegebene Lizenz gilt. *Vorsicht bei Änderungen der Lizenz die zu restriktiveren Bedingungen führen!*
`oparlSince`	date-time	Zeitpunkt, ab dem OParl für dieses Body bereitgestellt wurde. Dies hilft, um die Datenqualität einzuschätzen, denn erst ab der Einrichtung für OParl kann sichergestellt werden, dass sämtliche Werte korrekt in der Original-Quelle vorliegen.
`ags`	string	Der achtstellige Amtliche Gemeindeschlüssel[24].
`rgs`	string	Der zwölfstellige Regionalschlüssel.
`equivalent`	array of url	Dient der Angabe zusätzlicher URLs, die dieselbe Körperschaft repräsentieren. Hier können beispielsweise der entsprechende Eintrag der gemeinsamen Normdatei der Deutschen Nationalbibliothek[25], der DBPedia[26] oder der Wikipedia[27] angegeben werden. Body- oder System-Objekte mit anderen OParl-Versionen **dürfen nicht** Teil der Liste sein.

[24] Amtliche Gemeindeschlüssel können im Gemeindeverzeichnis (GV-ISys) des Statistischen Bundesamtes eingesehen werden

[25] Gemeinsame Normdatei `http://www.dnb.de/gnd`

[26] DBPedia `http://www.dbpedia.org/`

[27] Wikipedia `http://de.wikipedia.org/`

Name	Typ	Beschreibung
`contactEmail`	string	Dient der Angabe einer Kontakt-E-Mail-Adresse. Die Adresse soll die Kontaktaufnahme zu einer für die Körperschaft und idealerweise das parlamentarische Informationssystem zuständigen Stelle ermöglichen.
`contactName`	string	Name oder Bezeichnung der mit `contactEmail` erreichbaren Stelle.
`organization`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Gruppierungen der Körperschaft.
`person`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Personen der Körperschaft.
`meeting`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Sitzungen der Körperschaft.
`paper`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Drucksachen der Körperschaft.
`legislativeTerm`	array of object (LegislativeTerm)	**ZWINGEND** Objektliste mit den Wahlperioden der Körperschaft.
`agendaItem`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Tagesordnungspunkten der Körperschaft. Neu in OParl 1.1.
`consultation`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Beratungen der Körperschaft. Neu in OParl 1.1.
`file`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Dateien der Körperschaft. Neu in OParl 1.1.
`locationList`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Ortsangaben der Körperschaft. Neu in OParl 1.1.
`legislativeTermList`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Legislaturperioden der Körperschaft. Neu in OParl 1.1. Die externe Objektliste enthält die gleichen Objekte wie `legislativeTerm`
`membership`	url (externalList)	**ZWINGEND** Link zur Objektliste mit allen Mitgliedschaften der Körperschaft. Neu in OParl 1.1.
`classification`	string	Art der Körperschaft.
`location`	object (Location)	Ort, an dem die Körperschaft beheimatet ist.
`mainOrganization`	url (Organization)	Das zentrale Gremium einer Körperschaft. Bei einer Stadt wäre das z.B. die Vollversammlung des Stadtrats, beim Bundestag das Plenum.
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/body/0",
    "type": "https://schema.oparl.org/1.1/Body",
    "system": "https://oparl.example.org/",
    "contactEmail": "ris@beispielstadt.de",
    "contactName": "RIS-Betreuung",
    "ags": "05315000",
    "rgs": "053150000000",
    "equivalent": [
        "http://d-nb.info/gnd/2015732-0",
        "http://dbpedia.org/resource/Cologne"
    ],
    "shortName": "Köln",
    "name": "Stadt Köln, kreisfreie Stadt",
    "website": "http://www.beispielstadt.de/",
    "license": "http://creativecommons.org/licenses/by/4.0/",
    "licenseValidSince": "2014-01-01T00:00:00+02:00",
    "organization": "https://oparl.example.org/body/0/organizations/",
    "person": "https://oparl.example.org/body/0/persons/",
    "meeting": "https://oparl.example.org/body/0/meetings/",
    "paper": "https://oparl.example.org/body/0/papers/",
    "agendaItem": "https://oparl.example.org/body/0/agendaItems/",
    "consultation": "https://oparl.example.org/body/0/consultations/",
    "file": "https://oparl.example.org/body/0/files/",
    "locationList": "https://oparl.example.org/body/0/locations/",
    "membership": "https://oparl.example.org/body/0/memberships/",
    "legislativeTerm": [
        {
            "id": "https://oparl.example.org/term/21",
            "type": "https://schema.oparl.org/1.1/LegislativeTerm",
            "body": "https://oparl.example.org/body/0",
            "name": "21. Wahlperiode",
            "startDate": "2010-12-03",
            "endDate": "2013-12-03",
            "created": "2014-01-08T14:28:31+01:00",
            "modified": "2014-01-08T14:28:31+01:00"
        }
    ],
    "location": {
        "id": "https://oparl.example.org/location/0",
        "type": "https://schema.oparl.org/1.1/Location",
        "description": "Rathaus der Beispielstadt, Ratshausplatz 1, 12345 Beispielstadt",
        "created": "2014-01-08T14:28:31+01:00",
        "modified": "2014-01-08T14:28:31+01:00",
        "geojson": {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    50.1234,
```

```
                    10.4321
                ]
            },
            "properties": {
                "name": "Rathausplatz"
            }
        }
    },
    "classification": "Kreisfreie Stadt",
    "created": "2014-01-08T14:28:31+01:00",
    "modified": "2014-01-08T14:28:31+01:00"
}
```

3.6 LegislativeTerm

Dieser Objekttyp dient der Beschreibung einer Wahlperiode.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`body`	url (Body)	Rückreferenz auf die Körperschaft, welche nur dann ausgegeben werden muss, wenn das LegislativeTerm-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`name`	string	Nutzerfreundliche Bezeichnung der Wahlperiode.
`startDate`	date	Der erste Tag der Wahlperiode.
`endDate`	date	Der letzte Tag der Wahlperiode.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

3.7 Organization

Dieser Objekttyp dient dazu, Gruppierungen von Personen abzubilden, die in der parlamentarischen Arbeit eine Rolle spielen. Dazu zählen in der Praxis insbesondere Fraktionen und Gremien.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`body`	url (Body)	Körperschaft, zu der diese Gruppierung gehört.
`name`	string	Offizielle (lange) Form des Namens der Gruppierung.
`membership`	array of url (Membership)	Mitgliedschaften dieser Gruppierung.
`meeting`	url (externalList)	URL auf eine externe Objektliste mit den Sitzungen dieser Gruppierung. Invers zur Eigenschaft `organization` der Klasse `oparl:Meeting`
`consultation`	url (externalList)	URL auf eine externe Objektliste mit den Beratungen dieser Gruppierung. Invers zur Eigenschaft `organization` der Klasse `oparl:Consultation`
`shortName`	string	Der Name der Gruppierung als Kurzform.
`post`	array of string	Positionen, die für diese Gruppierung vorgesehen sind.
`subOrganizationOf`	url (Organization)	URL einer eventuellen übergeordneten Gruppierung.

Name	Typ	Beschreibung
`organizationType`	string	Grobe Kategorisierung der Gruppierung. Mögliche Werte sind “Gremium”, “Partei”, “Fraktion”, “Verwaltungsbereich”, “externes Gremium”, “Institution” und “Sonstiges”.
`classification`	string	Die Art der Gruppierung. In Frage kommen z.B. “Parlament”, “Ausschuss”, “Beirat”, “Projektbeirat”, “Kommission”, “AG”, “Verwaltungsrat”, “Fraktion” oder “Partei”. Die Angabe **sollte** möglichst präzise erfolgen. Außerdem **sollten** Abkürzungen vermieden werden. Für die höchste demokratische Instanz in der Kommune **sollte** immer der Begriff “Parlament” verwendet werden, nicht “Rat” oder “Hauptausschuss”.
`startDate`	date	Gründungsdatum der Gruppierung. Kann z. B. das Datum der konstituierenden Sitzung sein.
`endDate`	date	Datum des letzten Tages der Existenz der Gruppierung.
`website`	url	Allgemeine Website der Gruppierung.
`location`	object (Location)	Ort, an dem die Organisation beheimatet ist
`externalBody`	url (Body)	Externer OParl Body, der dieser Organisation entspricht. Diese Eigenschaft ist dafür gedacht auf eventuelle konkretere OParl-Schnittstellen zu verweisen. Ein Beispiel hierfür wäre eine Stadt, die sowohl ein übergreifendes parlamentarisches Informationssystem, als auch bezirksspezifische Systeme hat.
`memberCount`	integer	Die Anzahl der Mitglieder des Gremiums, einschließlich der Mitglieder ohne Stimmrecht. Diese Zahl entspricht der rechtlich festgelegten Mitgliederzahl und damit nicht zwangsläufig der Anzahl der aktiven `Membership`-Objekten dieses Gremiums, da z.B. auch für Vertreter Mitgliedschaften angegeben seien können.
`votingMemberCount`	integer	Der Anzahl der stimmberechtigten Mitglieder. Diese Zahl entspricht nicht zwangsläufig der Anzahl der aktiven `Membership`-Objekten dieses Gremiums mit `"votingRight": true`, sondern der rechtlich festgelegten Zahl an stimmberechtigten Mitgliedern, die z.B. zur Bestimmung der Beschlussfähigkeit verwendet werden kann.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	

Name	Typ	Beschreibung
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/organization/34",
    "type": "https://schema.oparl.org/1.1/Organization",
    "body": "https://oparl.example.org/bodies/1",
    "name": "Ausschuss für Haushalt und Finanzen",
    "shortName": "Finanzausschuss",
    "startDate": "2012-07-17",
    "organizationType": "Gremium",
    "location": {
        "id": "https://oparl.example.org/location/0",
        "type": "https://schema.oparl.org/1.1/Location",
        "description": "Rathaus der Beispielstadt, Ratshausplatz 1, 12345 Beispielstadt",
        "created": "2012-01-06T12:01:00+01:00",
        "modified": "2012-01-08T14:05:27+01:00",
        "geojson": {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    50.1234,
                    10.4321
                ]
            },
            "properties": {
                "name": "Rathausplatz"
            }
        }
    },
    "post": [
        "Vorsitzender",
        "1. Stellvertreter",
        "Mitglied"
    ],
    "meeting": "https://oparl.example.org/organization/34/meetings",
    "membership": [
        "https://oparl.example.org/membership/27",
        "https://oparl.example.org/membership/48",
        "https://oparl.example.org/membership/57"
    ],
    "classification": "Ausschuss",
    "keyword": [
        "finanzen",
        "haushalt"
    ],
    "created": "2012-07-16T00:00:00+02:00",
```

```
    "modified": "2012-08-16T12:34:56+02:00"
}
```

3.8 Person

Jede natürliche Person, die in der parlamentarischen Arbeit tätig und insbesondere Mitglied in einer Gruppierung (oparl:Organization) ist, wird mit einem Objekt vom Typ `oparl:Person` abgebildet.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`body`	url (Body)	Körperschaft, zu der die Person gehört.
`name`	string	Der vollständige Name der Person mit akademischem Grad und dem gebräuchlichen Vornamen, wie er zur Anzeige durch den Client genutzt werden kann.
`familyName`	string	Familienname bzw. Nachname.
`givenName`	string	Vorname bzw. Taufname.
`formOfAddress`	string	Anrede.
`affix`	string	Namenszusatz (z.B. `jun.` oder `MdL.`)
`title`	array of string	Akademische Titel
`gender`	string	Geschlecht. Vorgegebene Werte sind `female` und `male`, weitere werden durch die durchgehend klein geschriebene englische Bezeichnung angegeben. Für den Fall, dass das Geschlecht der Person unbekannt ist, **sollte** die Eigenschaft nicht ausgegeben werden.
`phone`	array of string	Telefonnummern der Person.
`email`	array of string	E-Mail-Adressen der Person.
`location`	url (Location)	Referenz der Kontakt-Anschrift der Person.
`locationObject`	object (Location)	Kontakt-Anschrift der Person. Wenn diese Eigenschaft ausgegeben wird, dann **muss** auch die Eigenschaft `location` ausgegeben werden und auf das gleiche Location-Objekt verweisen. Dieses Feld sollte die eigentliche Ausgabeform von `location` in OParl 1.0 werden. vgl. https://github.com/OParl/spec/issues/373. Neu in OParl 1.1
`status`	array of string	Status, d.h. Rollen in der Kommune.
`membership`	array of object (Membership)	Mitgliedschaften der Person in Gruppierungen, z. B. Gremien und Fraktionen. Es **sollen** sowohl aktuelle als auch vergangene Mitgliedschaften angegeben werden
`image`	object (File)	Ein Bild der Person. Aus Kompatibilitätsgründen wird **empfohlen, nur die Bildformate JPEG und PNG zu verwenden. Es sollten keine Bilder ausgeliefert werden, die nur als Platzhalter gedacht sind.

Name	Typ	Beschreibung
`life`	string	Kurzer Informationstext zur Person. Eine Länge von weniger als 300 Zeichen ist **empfohlen**
`lifeSource`	string	Angabe der Quelle, aus der die Informationen für `life` stammen. Bei Angabe von `life` ist diese Eigenschaft **empfohlen**
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/person/29",
    "type": "https://schema.oparl.org/1.1/Person",
    "body": "https://oparl.example.org/body/0",
    "name": "Prof. Dr. Max Mustermann",
    "familyName": "Mustermann",
    "givenName": "Max",
    "title": [
        "Prof.",
        "Dr."
    ],
    "formOfAddress": "Ratsfrau",
    "gender": "male",
    "email": [
        "max@mustermann.de"
    ],
    "phone": [
        "+493012345678"
    ],
    "status": [
        "Bezirksbürgermeister"
    ],
    "membership": [
        {
            "id": "https://oparl.example.org/memberships/385",
            "type": "https://schema.oparl.org/1.1/Membership",
            "organization": "https://oparl.example.org/organizations/5",
            "role": "Vorsitzende",
            "votingRight": true,
            "startDate": "2013-12-03",
            "created": "2011-11-11T11:11:00+01:00",
            "modified": "2012-08-16T14:05:27+02:00"
        },
        {
```

```
            "id": "https://oparl.example.org/memberships/693",
            "type": "https://schema.oparl.org/1.1/Membership",
            "organization": "https://oparl.example.org/organizations/9",
            "role": "Sachkundige Bürgerin",
            "votingRight": false,
            "startDate": "2013-12-03",
            "endDate": "2014-07-28",
            "created": "2011-11-11T11:11:00+01:00",
            "modified": "2012-08-16T14:05:27+02:00"
        }
    ],
    "created": "2011-11-11T11:11:00+01:00",
    "modified": "2012-08-16T14:05:27+02:00"
}
```

3.9 Membership

Über Objekte dieses Typs wird die Mitgliedschaft von Personen in Gruppierungen dargestellt. Diese Mitgliedschaften können zeitlich begrenzt sein. Zudem kann abgebildet werden, dass eine Person eine bestimmte Rolle bzw. Position innerhalb der Gruppierung innehat, beispielsweise den Vorsitz einer Fraktion.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`person`	url (Person)	Rückreferenz auf Person, welches nur dann ausgegeben werden muss, wenn das Membership-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`organization`	url (Organization)	Die Gruppierung, in der die Person Mitglied ist oder war.
`role`	string	Rolle der Person für die Gruppierung. Kann genutzt werden, um verschiedene Arten von Mitgliedschaften zum Beispiel in Gremien zu unterscheiden.
`votingRight`	boolean	Gibt an, ob die Person in der Gruppierung stimmberechtigtes Mitglied ist.
`startDate`	date	Datum, an dem die Mitgliedschaft beginnt.
`endDate`	date	Datum, an dem die Mitgliedschaft endet.
`onBehalfOf`	url (Organization)	Die Gruppierung, für die die Person in der unter `organization` angegebenen Organisation sitzt. Beispiel: Mitgliedschaft als Vertreter einer Ratsfraktion, einer Gruppierung oder einer externen Organisation.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

3.10 Meeting

Eine Sitzung ist die Versammlung einer oder mehrerer Gruppierungen (oparl:Organization) zu einem bestimmten Zeitpunkt an einem bestimmten Ort.

Die geladenen Teilnehmer der Sitzung sind jeweils als Objekte vom Typ oparl:Person, die in entsprechender Form referenziert werden. Verschiedene Dateien (Einladung, Ergebnis- und Wortprotokoll, sonstige Anlagen) können referenziert werden. Die Inhalte einer Sitzung werden durch Tagesordnungspunkte (oparl:AgendaItem) abgebildet.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`name`	string	Name der Sitzung.

Name	Typ	Beschreibung
`meetingState`	string	Aktueller Status der Sitzung. **Empfohlen** ist die Verwendung von `terminiert` (geplant), `eingeladen` (vor der Sitzung bis zur Freigabe des Protokolls) und `durchgeführt` (nach Freigabe des Protokolls).
`cancelled`	boolean	Wenn die Sitzung ausfällt, wird cancelled auf true gesetzt.
`start`	date-time	Datum und Uhrzeit des Anfangszeitpunkts der Sitzung. Bei einer zukünftigen Sitzung ist dies der geplante Zeitpunkt, bei einer stattgefundenen **kann** es der tatsächliche Startzeitpunkt sein.
`end`	date-time	Endzeitpunkt der Sitzung als Datum/Uhrzeit. Bei einer zukünftigen Sitzung ist dies der geplante Zeitpunkt, bei einer stattgefundenen **kann** es der tatsächliche Endzeitpunkt sein.
`location`	object (Location)	Sitzungsort.
`organization`	array of url (Organization)	Gruppierungen, denen die Sitzung zugeordnet ist. Im Regelfall wird hier eine Gruppierung verknüpft sein, es kann jedoch auch gemeinsame Sitzungen mehrerer Gruppierungen geben. Das erste Element **sollte** dann das federführende Gremium sein.
`participant`	array of url (Person)	Personen, die an der Sitzung teilgenommen haben (d.h. nicht nur die eingeladenen Personen, sondern die tatsächlich anwesenden). Diese Eigenschaft kann selbstverständlich erst nach dem Stattfinden der Sitzung vorkommen.
`invitation`	object (File)	Einladungsdokument zur Sitzung.
`resultsProtocol`	object (File)	Ergebnisprotokoll zur Sitzung. Diese Eigenschaft kann selbstverständlich erst nachdem Stattfinden der Sitzung vorkommen.
`verbatimProtocol`	object (File)	Wortprotokoll zur Sitzung. Diese Eigenschaft kann selbstverständlich erst nach dem Stattfinden der Sitzung vorkommen.
`auxiliaryFile`	array of object (File)	Dateianhang zur Sitzung. Hiermit sind Dateien gemeint, die üblicherweise mit der Einladung zu einer Sitzung verteilt werden, und die nicht bereits über einzelne Tagesordnungspunkte referenziert sind.
`agendaItem`	array of object (AgendaItem)	Tagesordnungspunkte der Sitzung. Die Reihenfolge ist relevant. Es kann Sitzungen ohne TOPs geben.
`license`	string	
`keyword`	array of string	
`created`	date-time	

Name	Typ	Beschreibung
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/meeting/281",
    "type": "https://schema.oparl.org/1.1/Meeting",
    "name": "4. Sitzung des Finanzausschusses",
    "start": "2013-01-04T08:00:00+01:00",
    "end": "2013-01-04T12:00:00+01:00",
    "location": {
        "id": "https://oparl.example.org/location/0",
        "type": "https://schema.oparl.org/1.1/Location",
        "description": "Rathaus der Beispielstadt, Ratshausplatz 1, 12345 Beispielstadt",
        "created": "2012-01-06T12:01:00+01:00",
        "modified": "2012-01-08T14:05:27+01:00",
        "geojson": {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    50.1234,
                    10.4321
                ]
            },
            "properties": {
                "name": "Rathausplatz"
            }
        }
    },
    "organization": [
        "https://oparl.example.org/organization/34"
    ],
    "invitation": {
        "id": "https://oparl.example.org/files/57739",
        "type": "https://schema.oparl.org/1.1/File",
        "name": "Einladung",
        "fileName": "einladung.pdf",
        "mimeType": "application/pdf",
        "date": "2012-01-08",
        "modified": "2012-01-08T14:05:27+01:00",
        "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
        "size": 82930,
        "accessUrl": "https://oparl.example.org/files/57739.pdf",
        "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
        "created": "2012-01-06T12:01:00+01:00"
    },
    "resultsProtocol": {
```

```
    "id": "https://oparl.example.org/files/57739",
    "type": "https://schema.oparl.org/1.1/File",
    "name": "Protokoll",
    "fileName": "protokoll.pdf",
    "mimeType": "application/pdf",
    "date": "2012-01-08",
    "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
    "size": 82930,
    "accessUrl": "https://oparl.example.org/files/57739.pdf",
    "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
    "modified": "2012-01-08T14:05:27+01:00",
    "created": "2012-01-06T12:01:00+01:00"
},
"verbatimProtocol": {
    "id": "https://oparl.example.org/files/57739",
    "type": "https://schema.oparl.org/1.1/File",
    "name": "Wortprotokoll",
    "fileName": "wortprotokoll.pdf",
    "mimeType": "application/pdf",
    "date": "2012-01-08",
    "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
    "size": 82930,
    "accessUrl": "https://oparl.example.org/files/57739.pdf",
    "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
    "modified": "2012-01-08T14:05:27+01:00",
    "created": "2012-01-08T14:05:27+01:00"
},
"auxiliaryFile": [
    {
        "id": "https://oparl.example.org/files/57739",
        "type": "https://schema.oparl.org/1.1/File",
        "name": "Nachtrags-Tagesordnung",
        "fileName": "nachtrag-TO.pdf",
        "mimeType": "application/pdf",
        "date": "2012-01-08",
        "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
        "size": 82930,
        "accessUrl": "https://oparl.example.org/files/57739.pdf",
        "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
        "modified": "2012-01-08T14:05:27+01:00",
        "created": "2012-01-08T14:05:27+01:00"
    }
],
"agendaItem": [
    {
        "id": "https://oparl.example.org/agendaitem/3271",
        "type": "https://schema.oparl.org/1.1/AgendaItem",
        "meeting": "https://oparl.example.org/meeting/281",
        "number": "10.1",
        "name": "Satzungsänderung für Ausschreibungen",
        "public": true,
        "consultation": "https://oparl.example.org/consultation/1034",
```

```
            "result": "Geändert beschlossen",
            "resolutionText": "Der Beschluss weicht wie folgt vom Antrag ab: ...",
            "created": "2012-01-06T12:01:00+01:00",
            "modified": "2012-01-08T14:05:27+01:00"
        }
    ],
    "created": "2012-01-06T12:01:00+01:00",
    "modified": "2012-01-08T14:05:27+01:00"
}
```

3.11 AgendaItem

Tagesordnungspunkte sind die Bestandteile von Sitzungen (`oparl:Meeting`). Jeder Tagesordnungspunkt widmet sich inhaltlich einem bestimmten Thema, wozu in der Regel auch die Beratung bestimmter Drucksachen gehört.

Die Beziehung zwischen einem Tagesordnungspunkt und einer Drucksache wird über ein Objekt vom Typ `oparl:Consultation` hergestellt, das über die Eigenschaft `consultation` referenziert werden kann.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`meeting`	url (Meeting)	Rückreferenz auf das Meeting, welches nur dann ausgegeben werden muss, wenn das agendaItem-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`number`	string	Gliederungs-"Nummer" des Tagesordnungspunktes. Eine beliebige Zeichenkette, wie z. B. "10.", "10.1", "C", "c)" o. ä. Die Reihenfolge wird nicht dadurch, sondern durch die Reihenfolge der TOPs im `agendaItem`-Attribut von `oparl:Meeting` festgelegt, **sollte** allerdings zu dieser identisch sein.
`order`	integer	**ZWINGEND** Neu in OParl 1.1: Die Position des Tagesordnungspunkts in der Sitzung, wenn alle Tagesordnungspunkte von 0 an durchgehend numeriert werden. Diese Nummer entspricht der Position in `Meeting:agendaItem`
`name`	string	Das Thema des Tagesordnungspunktes.
`public`	boolean	Kennzeichnet, ob der Tagesordnungspunkt zur Behandlung in öffentlicher Sitzung vorgesehen ist/war. Es wird ein Wahrheitswert (`true` oder `false`) erwartet.
`consultation`	url (Consultation)	Beratung, die diesem Tagesordnungspunkt zugewiesen ist.
`result`	string	Kategorische Information darüber, welches Ergebnis die Beratung des Tagesordnungspunktes erbracht hat, in der Bedeutung etwa "Unverändert beschlossen" oder "Geändert beschlossen".
`resolutionText`	string	Falls in diesem Tagesordnungspunkt ein Beschluss gefasst wurde, kann hier ein Text angegeben werden. Das ist besonders dann in der Praxis relevant, wenn der gefasste Beschluss (z. B. durch Änderungsantrag) von der Beschlussvorlage abweicht.

Name	Typ	Beschreibung
`resolutionFile`	object (File)	Falls in diesem Tagesordnungspunkt ein Beschluss gefasst wurde, kann hier eine Datei angegeben werden. Das ist besonders dann in der Praxis relevant, wenn der gefasste Beschluss (z. B. durch Änderungsantrag) von der Beschlussvorlage abweicht.
`auxiliaryFile`	array of object (File)	Weitere Dateianhänge zum Tagesordnungspunkt.
`start`	date-time	Datum und Uhrzeit des Anfangszeitpunkts des Tagesordnungspunktes. Bei zukünftigen Tagesordnungspunkten ist dies der geplante Zeitpunkt, bei einem stattgefundenen **kann** es der tatsächliche Startzeitpunkt sein.
`end`	date-time	Endzeitpunkt des Tagesordnungspunktes als Datum/Uhrzeit. Bei zukünftigen Tagesordnungspunkten ist dies der geplante Zeitpunkt, bei einer stattgefundenen **kann** es der tatsächliche Endzeitpunkt sein.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

3.12 Paper

Dieser Objekttyp dient der Abbildung von Drucksachen in der parlamentarischen Arbeit, wie zum Beispiel Anfragen, Anträgen und Beschlussvorlagen. Drucksachen werden in Form einer Beratung (oparl:Consultation) im Rahmen eines Tagesordnungspunkts (oparl:AgendaItem) einer Sitzung (oparl:Meeting) behandelt.

Drucksachen spielen in der schriftlichen wie mündlichen Kommunikation eine besondere Rolle, da in vielen Texten auf bestimmte Drucksachen Bezug genommen wird. Hierbei kommen in parlamentarischen Informationssystemen in der Regel unveränderliche Kennungen der Drucksachen zum Einsatz.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`body`	url (Body)	Körperschaft, zu der die Drucksache gehört.
`name`	string	Titel der Drucksache.
`reference`	string	Kennung bzw. Aktenzeichen der Drucksache, mit der sie in der parlamentarischen Arbeit eindeutig referenziert werden kann.
`date`	date	Datum, welches als Startpunkt für Fristen u.ä. verwendet ist.
`paperType`	string	Art der Drucksache, z. B. Beantwortung einer Anfrage.

Name	Typ	Beschreibung
`relatedPaper`	array of url (Paper)	Inhaltlich verwandte Drucksachen.
`superordinatedPaper`	array of url (Paper)	Übergeordnete Drucksachen.
`subordinatedPaper`	array of url (Paper)	Untergeordnete Drucksachen.
`mainFile`	object (File)	Die Hauptdatei zu dieser Drucksache. Beispiel: Die Drucksache repräsentiert eine Beschlussvorlage und die Hauptdatei enthält den Text der Beschlussvorlage. Sollte keine eindeutige Hauptdatei vorhanden sein, wird diese Eigenschaft nicht ausgegeben.
`auxiliaryFile`	array of object (File)	Alle weiteren Dateien zur Drucksache ausgenommen der gegebenenfalls in `mainFile` angegebenen.
`location`	array of object (Location)	Sofern die Drucksache einen inhaltlichen Ortsbezug hat, beschreibt diese Eigenschaft den Ort in Textform und/oder in Form von Geodaten.
`originatorPerson`	array of url (Person)	Urheber der Drucksache, falls der Urheber eine Person ist. Es können auch mehrere Personen angegeben werden.
`underDirectionOf`	array of url (Organization)	Federführung. Amt oder Abteilung, für die Inhalte oder Beantwortung der Drucksache verantwortlich.
`originatorOrganization`	array of url (Organization)	Urheber der Drucksache, falls der Urheber eine Gruppierung ist. Es können auch mehrere Gruppierungen angegeben werden.
`consultation`	array of object (Consultation)	Beratungen der Drucksache.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/paper/749",
    "type": "https://schema.oparl.org/1.1/Paper",
    "body": "https://oparl.example.org/bodies/1",
    "name": "Antwort auf Anfrage 1200/2014",
    "reference": "1234/2014",
    "date": "2014-04-04",
    "paperType": "Beantwortung einer Anfrage",
    "relatedPaper": [
        "https://oparl.example.org/paper/699"
    ],
    "mainFile": {
        "id": "https://oparl.example.org/files/57737",
        "type": "https://schema.oparl.org/1.1/File",
```

```
    "name": "Anlage 1 zur Anfrage",
    "fileName": "anlage_1_zur_anfrage.pdf",
    "mimeType": "application/pdf",
    "date": "2013-01-04",
    "sha1Checksum": "d749751af44a32c818b9b1e1515251c67734f5d2",
    "size": 82930,
    "accessUrl": "https://oparl.example.org/files/57737.pdf",
    "downloadUrl": "https://oparl.example.org/files/download/57737.pdf",
    "license": "http://www.opendefinition.org/licenses/cc-by",
    "created": "2013-01-04T07:54:13+01:00",
    "modified": "2013-01-04T07:54:13+01:00"
},
"auxiliaryFile": [
    {
        "id": "https://oparl.example.org/files/57739",
        "type": "https://schema.oparl.org/1.1/File",
        "name": "Anlage 1 zur Anfrage",
        "fileName": "anlage.pdf",
        "mimeType": "application/pdf",
        "date": "2013-01-04",
        "sha1Checksum": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
        "size": 82930,
        "accessUrl": "https://oparl.example.org/files/57739.pdf",
        "downloadUrl": "https://oparl.example.org/files/download/57739.pdf",
        "text": "Der Übersichtsplan zeigt alle Ebenen des ...",
        "masterFile": "https://oparl.example.org/files/57738",
        "license": "http://www.opendefinition.org/licenses/cc-by",
        "created": "2013-01-04T07:54:13+01:00",
        "modified": "2013-01-04T07:54:13+01:00"
    }
],
"location": [
    {
        "id": "https://oparl.example.org/locations/29856",
        "type": "https://schema.oparl.org/1.1/Location",
        "description": "Honschaftsstraße 312, Köln",
        "created": "2012-01-08T14:05:27+01:00",
        "modified": "2012-01-08T14:05:27+01:00",
        "geojson": {
            "type": "Point",
            "coordinates": [
                7.03291,
                50.98249
            ]
        }
    }
],
"originatorPerson": [
    "https://oparl.example.org/person/2000",
    "https://oparl.example.org/person/1000"
],
"originatorOrganization": [
```

```
        "https://oparl.example.org/organization/2000",
        "https://oparl.example.org/organization/1000"
    ],
    "consultation": [
        {
            "id": "https://oparl.example.org/consultation/47594",
            "type": "https://schema.oparl.org/1.1/Consultation",
            "agendaItem": "https://oparl.example.org/agendaitem/15569",
            "meeting": "https://oparl.example.org/meeting/243",
            "organization": [
                "https://oparl.example.org/organization/96"
            ],
            "authoritative": false,
            "role": "Beschlussfassung",
            "created": "2012-01-08T14:05:27+01:00",
            "modified": "2012-01-08T14:05:27+01:00"
        }
    ],
    "underDirectionOf": [
        "https://oparl.example.org/organization/2000"
    ],
    "created": "2013-01-08T12:05:27+01:00",
    "modified": "2013-01-08T12:05:27+01:00"
}
```

3.13 Consultation

Der Objekttyp `oparl:Consultation` dient dazu, die Beratung einer Drucksache (`oparl:Paper`) in einer Sitzung abzubilden. Dabei ist es nicht entscheidend, ob diese Beratung in der Vergangenheit stattgefunden hat oder diese für die Zukunft geplant ist. Die Gesamtheit aller Objekte des Typs `oparl:Consultation` zu einer bestimmten Drucksache bildet das ab, was in der Praxis als "Beratungsfolge" der Drucksache bezeichnet wird.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`paper`	url (Paper)	Referenz auf das Paper, welche nur dann ausgegeben werden muss, wenn das Consultation-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`agendaItem`	url (AgendaItem)	Referenz auf den Tagesordnungspunkt, unter dem die Drucksache beraten wird, welcher nur dann ausgegeben werden muss, wenn das Consultation-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`meeting`	url (Meeting)	Referenz auf die Sitzung, in der die Drucksache beraten wird oder wurde, welche nur dann ausgegeben werden muss, wenn das Consultation-Objekt einzeln abgerufen wird, d.h. nicht Teil einer internen Ausgabe ist.
`organization`	array of url (Organization)	Gremium, in dem die Drucksache beraten wird. Hier kann auch eine mit Liste von Gremien angegeben werden (die verschiedenen `oparl:Body` und `oparl:System` angehören können). Die Liste ist dann geordnet. Das erste Gremium der Liste ist federführend.
`authoritative`	boolean	Drückt aus, ob bei dieser Beratung ein Beschluss zu der Drucksache gefasst wird oder wurde (`true`) oder nicht (`false`).
`role`	string	Rolle oder Funktion der Beratung. Zum Beispiel Anhörung, Entscheidung, Kenntnisnahme, Vorberatung usw.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

3.14 File

Ein Objekt vom Typ `oparl:File` repräsentiert eine Datei, beispielsweise eine PDF-Datei, ein RTF-Dokument oder ein Bild. Es hält Metadaten zu der Datei sowie URLs zum Zugriff auf die Datei bereit.

Objekte vom Typ `oparl:File` können unter anderem mit Drucksachen (`oparl:Paper`) oder Sitzungen (`oparl:Meeting`) in Beziehung stehen. Dies wird durch die Eigenschaft `paper` bzw. `meeting` angezeigt. Mehrere Objekte vom Typ `oparl:File` können mit einander in direkter Beziehung stehen, z.B. wenn sie den selben Inhalt in unterschiedlichen technischen Formaten wiedergeben. Hierfür werden die Eigenschaften `masterFile` bzw. `derivativeFile` eingesetzt. Das gezeigte Beispiel-Objekt repräsentiert eine PDF-Datei (zu erkennen an der Eigenschaft `mimeType`) und zeigt außerdem über die Eigenschaft `masterFile` an, von welcher anderen Datei es abgeleitet wurde. Umgekehrt **kann** über die Eigenschaft `derivativeFile` angezeigt werden, welche Ableitungen einer Datei existieren.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`name`	string	Ein zur Anzeige für Endnutzer bestimmter Name für dieses Objekt. Leerzeichen **dürfen** enthalten sein, Datei-Endungen wie ".pdf" **sollten nicht** enthalten sein.
`fileName`	string	Dateiname, unter dem die Datei in einem Dateisystem gespeichert werden kann. Beispiel: "einedatei.pdf". Da der Name den kompletten Unicode-Zeichenumfang nutzen kann, **sollten** Clients ggfs. selbst dafür sorgen, diesen beim Speichern in ein Dateisystem den lokalen Erfordernissen anzupassen.
`mimeType`	string	MIME-Type der Datei [28].
`date`	date	Datum, welches als Startpunkt für Fristen u.ä. verwendet ist.
`size`	integer	Größe der Datei in Bytes.
`sha1Checksum`	string	[Veraltet] SHA1-Prüfsumme des Dateiinhalts in Hexadezimal-Schreibweise. Sollte nicht mehr verwendet werden, da sha1 als unsicher gilt. Stattdessen sollte `sha512checksum` verwendet werden.
`sha512Checksum`	string	SHA512-Prüfsumme des Dateiinhalts in Hexadezimal-Schreibweise.
`text`	string	Reine Text-Wiedergabe des Dateiinhalts, sofern dieser in Textform wiedergegeben werden kann.
`accessUrl`	url	**ZWINGEND** URL zum allgemeinen Zugriff auf die Datei. Näheres unter Dateizugriffe.
`downloadUrl`	url	URL zum Download der Datei. Näheres unter Dateizugriffe.
`externalServiceUrl`	url	Externe URL, welche eine zusätzliche Zugriffsmöglichkeit bietet. Beispiel: YouTube-Video.
`masterFile`	url (File)	Datei, von der das aktuelle Objekt abgeleitet wurde. Details dazu in der allgemeinen Beschreibung weiter oben.

[28] vgl. RFC2046: http://tools.ietf.org/html/rfc2046

Name	Typ	Beschreibung
`derivativeFile`	array of url (File)	Dateien, die von dem aktuellen Objekt abgeleitet wurden. Details dazu in der allgemeinen Beschreibung weiter oben.
`fileLicense`	url	Lizenz, unter der die Datei angeboten wird. Wenn diese Eigenschaft nicht verwendet wird, ist der Wert von `license` beziehungsweise die Lizenz eines übergeordneten Objektes maßgeblich. Siehe license
`meeting`	array of url (Meeting)	Rückreferenzen auf Meeting-Objekte. Wird nur dann ausgegeben, wenn das File-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`agendaItem`	array of url (AgendaItem)	Rückreferenzen auf AgendaItem-Objekte. Wird nur dann ausgegeben, wenn das File-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`person`	url (Person)	Rückreferenz auf die Person, deren Bild die Datei darstellt. Wird nur dann ausgegeben, wenn das File-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`paper`	array of url (Paper)	Rückreferenzen auf Paper-Objekte. Wird nur dann ausgegeben, wenn das File-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel


```
{
    "id": "https://oparl.example.org/files/57737",
    "type": "https://schema.oparl.org/1.1/File",
    "name": "Anlage 1 zur Anfrage",
    "fileName": "anlage_1_zur_anfrage.pdf",
    "mimeType": "application/pdf",
    "date": "2013-01-04",
    "size": 82930,
    "sha1Checksum": "d749751af44a32c818b9b1e1515251c67734f5d2",
    "accessUrl": "https://oparl.example.org/files/57737.pdf",
    "downloadUrl": "https://oparl.example.org/files/download/57737.pdf",
    "derivativeFile": [
        "https://oparl.example.org/files/57739"
    ],
    "fileLicense": "http://www.opendefinition.org/licenses/cc-by",
    "created": "2013-01-04T07:54:13+01:00",
```

```
    "modified": "2013-01-04T07:54:13+01:00"
}
```

3.15 Location

Dieser Objekttyp dient dazu, einen Ortsbezug formal abzubilden. Ortsangaben können sowohl aus Textinformationen bestehen (beispielsweise dem Namen einer Straße/eines Platzes oder eine genaue Adresse) als auch aus Geodaten. Ortsangaben sind auch nicht auf einzelne Positionen beschränkt, sondern können eine Vielzahl von Positionen, Flächen, Strecken etc. abdecken.

Name	Typ	Beschreibung
`id`	url	
`type`	string	
`description`	string	Textuelle Beschreibung eines Orts, z. B. in Form einer Adresse.
`geojson`	object	Geodaten-Repräsentation des Orts. Der Wert dieser Eigenschaft **muss** der Spezifikation von GeoJSON entsprechen, d.h. es **muss** ein vollständiges `Feature`-Objekt ausgegeben werden.
`streetAddress`	string	Straße und Hausnummer der Anschrift.
`room`	string	Raumangabe der Anschrift
`postalCode`	string	Postleitzahl der Anschrift.
`subLocality`	string	Untergeordnete Ortsangabe der Anschrift, z.B. Stadtbezirk, Ortsteil oder Dorf.
`locality`	string	Ortsangabe der Anschrift.
`bodies`	array of url (Body)	Rückreferenzen auf Body-Objekte. Wird nur dann ausgegeben, wenn das Location-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`organizations`	array of url (Organization)	Rückreferenzen auf Organization-Objekte. Wird nur dann ausgegeben, wenn das Location-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`persons`	array of url (Person)	Rückreferenzen auf Person-Objekte. Wird nur dann ausgegeben, wenn das Location-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`meetings`	array of url (Meeting)	Rückreferenzen auf Meeting-Objekte. Wird nur dann ausgegeben, wenn das Location-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`papers`	array of url (Paper)	Rückreferenzen auf Paper-Objekte. Wird nur dann ausgegeben, wenn das Location-Objekt nicht als eingebettetes Objekt aufgerufen wird.
`license`	string	
`keyword`	array of string	
`created`	date-time	
`modified`	date-time	
`web`	url	
`deleted`	boolean	

Beispiel

```
{
    "id": "https://oparl.example.org/location/0",
    "type": "https://schema.oparl.org/1.1/Location",
    "description": "Deutscher Bundestag, Platz der Republik 1, 11011 Berlin",
    "streetAddress": "Platz der Republik 1",
    "postalCode": "11011",
    "locality": "Berlin",
    "room": "Plenarsaal",
    "geojson": {
        "type": "Feature",
        "geometry": {
            "type": "Point",
            "coordinates": [
                52.518855,
                13.376198
            ]
        },
        "properties": {
            "name": "Reichstagsgebäude"
        }
    }
}
```

4 Änderungen und Migration

Dieses Kapitel beschreibt die Änderungen zwischen den einzelnen OParl-Versionen und die zur Migration notwendigen Schritte.

4.1 OParl 1.1

In OParl 1.1 setzen wir die seit der Veröffentlichung von OParl 1.0 gewonnen Erfahrungen um. OParl 1.1 ist dabei im Sinne von semver kompatibel zu OParl 1.0. Das bedeutet, dass ein für OParl 1.0 entwickelter Client auch die Ausgabe von OParl 1.1 versteht. Dadurch wird der Migrationsaufwand von OParl 1.0 zu OParl 1.1 gering gehalten.

OParl 1.0 wurde in der Annahme geschrieben, dass für sechs Objekttypen (LegislativeTerm, Membership, AgendaItem, Consultation, File, Location) keine verlässlichen Werte für `created` und `modified` existieren. Aus diesem Grund hatten wir uns für das Design mit eingebetteten Objekten entschieden. Da sich nun jedoch herausgestellt hat, dass `created` und `modified` bei allen Objekten existieren, können auch für alle Objekte Listen angeboten werden. Das bringt bei große Vereinfachungen für Clients bei der Synchronisation.

Konkret sind `created` und `modified` in OParl 1.1 für alle Objekte zwingend und es gibt sechs neue externe Objektlisten in Body: AgendaItem, Consultation, File, LegislativeTerm, Location und Membership. Das Attribut für die Location-Liste in Body heißt dabei `locationList`, um eine Kollision mit dem bereits existierenden `location` zu vermeiden. Das gleiche gilt auch für `legislativeTermList`.

Es entsteht dabei Redundanz zwischen den bereits existierenden Objektlisten mit eingebetteten Objekten (Body, Paper, Meeting, Person, Organization) und den neuen externen Listen, die die bisher eingebetteten Objekte extern ausgeben. Diese Redundanz lässt sich auf Grund der Semver-Regeln in Version 1.1 nicht vermeiden und kann erst in einer Version 2 beseitigt werden. Um diese Redundanz zumindest bei der Aktualisierung eines lokalen Datenbestands vermeiden zu können wurde die URL-Parameter `omit_internal` eingeführt.

4.1.1 Weitere Änderungen

- Namespace-URLs werden durchgängig im Camel Case geschrieben
- Externe Objektlisten können ein `web` Attribut angeben
- Jedes Objekt des Schemas hat seine eigene Datei bekommen
- Definition eines Fehlerobjektes für die Ausnahmebehandlung
- sha1 veraltet und sha512 als Ersatz hinzugefügt. (s. https://shattered.io)
- Die Rückreferenz von Location auf Person wird zusätzlich auch noch eingebettet ausgeben (s. https://github.com/OParl/spec/issues/373)

4.2 OParl Next

- `Person` hat ein Feld `image` erhalten.
- `Body` kann `mainOrganization` angeben.
- Für `Organization` kann `memberCount` und `votingMemberCount` angegeben werden.